Journal of Artificial Intelligence Research 52 (2015) 507-542 Submitted 11/14; published 04/15

Weighted Electoral Control

Piotr Faliszewski FALISZEW@AGH.EDU.PL
Department of Computer Science
AGH University of Science and Technology
Krakow, Poland

Edith Hemaspaandra EH@CS.RIT.EDU
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623, USA

Lane A. Hemaspaandra LANE@CS.ROCHESTER.EDU
Department of Computer Science
University of Rochester
Rochester, NY 14627, USA

Abstract

Although manipulation and bribery have been extensively studied under weighted voting, there has been almost no work done on election control under weighted voting. This is unfortunate, since weighted voting appears in many important natural settings. In this paper, we study the complexity of controlling the outcome of weighted elections through adding and deleting voters. We obtain polynomial-time algorithms, NP-completeness results, and for many NP-complete cases, approximation algorithms. In particular, for scoring rules we completely characterize the complexity of weighted voter control. Our work shows that for quite a few important cases, either polynomial-time exact algorithms or polynomial-time approximation algorithms exist.

1. Introduction

In many real-world election systems the voters come with weights. Examples range from stockholder elections weighted by shares, to the US Electoral College, to the often-used example of the Nassau County Board of Supervisors, to (in effect) any parliamentary system in which the parties typically vote as blocks, to Sweden's system of wealth-weighted voting instituted in 1866 (and no longer used) where "the wealthiest members of the rural communities received as many as 5,000 votes" and "in 10 percent of the districts the weighted votes of just three voters could be decisive" (Congleton, 2011). Furthermore, there are many important voting applications within multiagent systems, e.g., in recommender systems (Ghosh, Mundhe, Hernandez, & Sen, 1999; Lu & Boutilier, 2011), planning (Ephrati & Rosenschein, 1997), and web search (Dwork, Kumar, Naor, & Sivakumar, 2001). In these applications, it is quite natural for the voters (i.e., agents) to be weighted (e.g., by the amount of trust we put in them or by the power of the resources they possess).

So it is not surprising that in the study of manipulative attacks on elections, weighted voting has been given great attention. For bribery and manipulation, two of the three most studied types of manipulative attacks on elections, study of the case of weighted voters has been extensively conducted. Yet for the remaining one of the three most studied types of attacks on elections, so-called control attacks, almost no attention has been given to the case of weighted voting; to the best of our knowledge, the only time this issue has been previously raised is in two M.S./Ph.D. theses (Russell, 2007; Lin, 2012). This lack of attention is troubling, since the key types of control attacks, such as adding and deleting voters, certainly do occur in many weighted elections. As a coda to this section, we will give some examples.

We study the complexity in weighted elections of arguably the most important types of control—adding and deleting voters—for various election systems. We focus on scoring rules, families of scoring rules, Condorcet-consistent rules, and weakCondorcet-consistent rules. Control by deleting (adding) voters asks whether in a given election a given candidate can be made to win by deleting (adding) at most a certain number of the voters (at most a certain number of the members of the pool of potential additional voters). These control types model issues that are found in many electoral settings, ranging from human to electronic. They are (abstractions of) issues often faced by people seeking to steer an election, such as experts doing campaign management, and deciding for example which k people to offer rides to the polls. Adding and deleting voters also can occur in multiagent systems. For example, if agents are entities over the Internet, then one can attempt a denial-of-service attack on some of them to prevent their votes from arriving on time. On the other hand, adding voters pertains to simply encouraging some agents to vote, multiplying the existing agents, or performing false-name attacks (for false-name attacks in related settings, see, for example, Wagman & Conitzer, 2014; Waggoner, Xia, & Conitzer, 2012; Aziz, Bachrach, Elkind, & Paterson, 2011).

Control was introduced (without weights) in 1992 in the seminal paper by Bartholdi, Tovey, and Trick (1992). Control has been the subject of much attention since. That attention, and the present paper, are part of the line of work, started by Bartholdi, Tovey, and Trick (1989, 1992) and Bartholdi and Orlin (1991), that seeks to determine for which types of manipulative attacks on elections the attacker's task requires just polynomial-time computation. For a more detailed discussion of this line of work, we refer the reader to the related work section at the end of the paper and to the surveys of Faliszewski, Hemaspaandra, Hemaspaandra, and Rothe (2009), Faliszewski, Hemaspaandra, and Hemaspaandra (2010), and Brandt, Conitzer, and Endriss (2013).

Our main results are as follows (see Section 6 for tables summarizing our results). First, in Section 4.1 we provide a detailed study of the complexity of voter control under scoring protocols, for the case of fixed numbers of candidates. We show that both constructive control by adding voters and constructive control by deleting voters are in P for t-approval (and so this also covers plurality and t'-veto[1]) and are NP-complete otherwise. It is interesting to compare this result to an analogous theorem regarding weighted coalitional manipulation: There are cases where the complexities of voter control and manipulation are the same (e.g., for plurality or for Borda) but there are also cases where voter control is easier (t-approval for $t \geq 2$, for elections with more than t candidates). Is it ever possible that weighted voter control is harder than weighted voting manipulation? We show that weighted voter control is NP-hard for Condorcet-consistent rules with at least three candidates (and so clearly is also NP-hard for weakCondorcet-consistent rules with at least

1. If the number of candidates is fixed, then t-veto can be expressed as $(m - t)$-approval, where m is the number of candidates. If the number of candidates is unbounded, then t-veto is not t'-approval.

three candidates). Since weighted coalitional manipulation for the 3-candidate Llull system is in P (Faliszewski, Hemaspaandra, & Schnoor, 2008), together with the fact that Llull is weakCondorcet-consistent, this implies that there is a setting where weighted voter control is harder than weighted coalitional manipulation.

In Sections 4.2 and 4.3 we focus on the complexity of weighted voter control under t-approval and t-veto, for the case of unbounded numbers of candidates. At the start of Section 4.2, we will explain why these are the most interesting cases. In Section 4.2 we resolve six problems left open by Lin (2012). We establish the complexity of weighted control by adding voters for 2-approval, 2-veto, and 3-approval, and of weighted control by deleting voters for 2-approval, 2-veto, and 3-veto. In Section 4.3, we give polynomial-time approximation algorithms for weighted voter control under t-approval and t-veto. Our algorithms seek to minimize the number of voters that are added or deleted.

We believe that the complexity of weighted voter control, and more generally the complexity of attacks on weighted elections, is an important and interesting research direction that deserves much further study. In particular, our research suggests that it is worthwhile to seek approximation algorithms for weighted elections problems and that doing so can lead to interesting algorithms.

2. Motivation for Studying Control in Weighted Elections

In the Introduction, we noted the importance of weights in many electoral settings and described the natural importance of—and gave pointers to the extensive line of work studying—control attacks in many (unweighted) settings. We also stated that control attacks can naturally be expected to occur even in many weighted election settings.

In the present section, we give some examples motivating the study of *weighted* electoral control.

Let us consider an academic department that has as its salient issue in a particular term the question of what course to add to its B.S. major's requirements. Suppose the department is highly polarized on this issue by research area, i.e., all the faculty in a given research area will vote as a block (either because they all agree, or because by tradition they meet before the actual faculty meetings to, within their group, reach a group position that they will all support). And suppose that for each group there are days/times where the entire group would be unlikely to attend, e.g., because that time is the time of the major yearly research conference in that area. The department chair, knowing that he or she has the power to schedule when faculty meetings are held, and what the agenda is at each meeting, might well model his or her task as a weighted control by deleting voters problem, in which the voters are the groups, each group's weight is the number of faculty members in the groups, and the deletion limit is one.

In fact, more generally, individual voters may blur into a collection of weighted votes in settings where the voter set partitions into groups that will express identical preferences. As another example of this sort, at one of the authors' schools, members of the faculty senate are chosen by the election system known as single transferable vote. However, it is not unheard of for departmental leaders to send out a friendly suggestion to the department's members regarding for whom to vote. If one assumes that the departments vote as blocks, and one is trying to decide which candidates to add to the election, or convince not to run in the election, in order to make a given candidate win (or not win), one in effect is studying weighted constructive control by adding candidates, weighted constructive control by deleting candidates, weighted destructive control by adding candidates, or weighted destructive control by deleting candidates.

The examples just given were ones where the weightings are created by individuals forming themselves into blocks, and that can occur even in highly political contexts. For example, in the US House of Representatives, on issues (for example, water rights or farm subsidies) on which a state's delegation tends to vote as a block in the parochial interests of its state's constituents or companies, pressure by lobbyists on state delegations to abstain from a given vote is in effect (give or take issues of failing to have a quorum) a control by deleting voters attack.

However, there are also many voting cases where the weights are inherent in the stand-alone individual voters, and in many of these cases control attacks may well occur. For example, consider US Corporate Elections. In these, the vote of each stockholder is weighted by his or her number of shares. The most natural way to frame control problems in this setting is the case of adding and deleting candidates, for example, regarding who is running for a spot as a company officer or director. But even voter control can come into play here, for example, through some actor sending mailings to—or phoning and speaking to—voters to convince them to abstain from voting, or to encourage voters to vote in this election. (The "bound on additions/deletions" model that counts number of voters, rather than their weights, is quite reasonable in this setting, because regardless of his or her weight, a given targeted voter can be addressed by, for example, a mailing/visit/phone-call, although in reality one admittedly might focus more resources on the biggest stockholders.)

There are many other weighted control examples we have not presented. Let us finish with what is an extremely high-stakes example. In the US Electoral College, which works by majority rule among the electors, the electors from each state usually vote as a block, since the system lets whoever has the greatest popular vote in the given state select every elector from that state (note: two of the fifty states have different policies). Thus the issue of, for example, whether someone such as Ralph Nader does or does not run, or does or does not withdraw his name from consideration at some particular time, can have a sweeping effect on the nation, in what is in effect a weighted control by adding/deleting candidates scenario.

We have now given a number of examples, some for voter control and some for candidate control, of settings where weighted control may occur. The examples we have given vary in their naturalness, and for both weighted and unweighted control there certainly are points on which the models don't capture all the nuances of the real world. For example, some electoral partitioning problems have geographic/contiguity constraints, groups that are modeled as voting as blocks may in fact have defectors, and Internet denial-of-service attacks may not have the freedom to suppress each vote independently but rather may have to suppress all or none of the votes coming over a given line/provider (see, e.g., Chen, Faliszewski, Niedermeier, & Talmon, 2014). Nonetheless, our belief is that the importance of weighted elections and the importance of control attacks should not remain forever separate. We feel that control attacks are sufficiently natural in many weighted settings—varying from academic departments to companies' stockholders to nations—that studying weighted control is worth undertaking. We also feel that, although this is not the subject of the present paper, it will be important for experimental studies to be undertaken to see the extent to which heuristic approaches can circumvent worst-case hardness results regarding weighted control (see Rothe & Schend, 2013, for an assessment of this type of approach in other settings, although see also Hemaspaandra & Williams, 2012, for a discussion of some limitations of heuristic attacks).

3. Preliminaries

We assume that the reader is familiar with the basic notions of computational complexity theory and the theory of algorithms. Below we provide relevant definitions and conventions regarding elections, election rules, and control in elections. We also review some NP-complete problems that we use in our reductions.

3.1 Elections

We take an election to be a pair $E = (C, V)$, where C is a set of candidates and V is a collection of voters. Each voter has a preference order over the set C. A *preference order* is a total, linear order that ranks the candidates from the most preferred one to the least preferred one. For example, if $C = \{a, b, c\}$ and some voter likes a best, then b, and then c, then his or her preference order is $a > b > c$. In weighted elections, each voter v also has a positive integer weight $\omega(v)$. A voter of weight $\omega(v)$ is treated by the election system as $\omega(v)$ unweighted voters. Given two collections of voters, V and W, we write $V + W$ to denote their concatenation.

3.2 Election Rules

An election rule (or voting rule) is a function R that given an election $E = (C, V)$ returns a subset $R(E) \subseteq C$, namely those candidates that are said to win the election.

An m-candidate scoring rule is defined through a nonincreasing vector $\alpha = (\alpha_1, \dots, \alpha_m)$ of nonnegative integers. For each voter v, each candidate c receives $\alpha_{pos(v,c)}$ points, where $pos(v, c)$ is the position of c in v's preference order. The candidates with the maximum total score are the winners. Given an election E and a voting rule R that assigns scores to the candidates, we write $score_E(c)$ to denote c's total score in E under R. The voting rule used will always be clear from context. Many election rules are defined through families of scoring rules, with one scoring vector for each possible number of candidates. For example:

1. Plurality rule uses vectors of the form $(1, 0, \dots, 0)$.
2. t-approval uses vectors $(\alpha_1, \dots, \alpha_m)$, where $\alpha_i = 1$ for each $i \in \{1, \dots, t\}$, and $\alpha_i = 0$ for $i > t$. By t-veto we mean the system that for m candidates uses the $(m-t)$-approval scoring vector. For m-candidate t-approval and t-veto systems we will often treat each vote as a 0/1 m-dimensional approval vector that indicates which candidates receive points from the vote. Naturally, such a vector contains exactly t ones for t-approval and exactly t zeroes for t-veto.[2]

2. We emphasize that such a view of t-approval and t-veto is correct in settings where the set of candidates remains fixed. If the set of candidates were to change (e.g., as in control by adding/deleting candidates), then we would have to use the standard, preference-order-based definition.

3. Borda's rule uses vectors of the form $(m-1, m-2, \ldots, 0)$, where m is the number of candidates.

Given an election $E = (C, V)$, a candidate c is a Condorcet winner (weak Condorcet winner) if for every other candidate $d \in C - \{c\}$ it holds that more than half (at least half) of the voters prefer c to d. Note that it is possible that there is no Condorcet winner in a given election, and it is even possible that there is no weak Condorcet winner in a given election. Let Condorcet denote the election system whose winner set is exactly the set of Condorcet winners, and let weakCondorcet denote the election system whose winner set is exactly the set of weak Condorcet winners. We say that a rule R is Condorcet-consistent if whenever there is a Condorcet winner he or she is the sole winner elected under R. Analogously, a rule is weakCondorcet-consistent if it elects exactly the weak Condorcet winners whenever they exist. Every weakCondorcet-consistent system is Condorcet-consistent, but the converse does not always hold.

There are many Condorcet-consistent rules. We will briefly touch upon the Copeland family of rules and the Maximin rule. For a given election $E = (C, V)$ and two distinct candidates $c, d \in C$, we let $N_E(c, d)$ be the number of voters that prefer c to d. Let α be a rational number, $0 \leq \alpha \leq 1$. Under Copeland$^\alpha$ the score of candidate $c \in C$ is defined as:

$$\|\{d \in C - \{c\} \mid N_E(c, d) > N_E(d, c)\}\| + \alpha\|\{d \in C - \{c\} \mid N_E(c, d) = N_E(d, c)\}\|,$$

and under Maximin the score of candidate $c \in C$ is defined as $\min_{d \in C - \{c\}} N_E(c, d)$. The candidates with the highest score are winners. Llull is another name for Copeland^1. Clearly, Llull and Maximin are weakCondorcet-consistent.

3.3 Electoral Control

We focus on constructive control by adding/deleting voters in weighted elections. However, there are also other standard types of control studied in the literature (e.g., control by adding/deleting candidates and various forms of partitioning of candidates and voters; we point the reader to Section 5 for a discussion of related work).

Definition 3.1. *Let R be a voting rule. In both weighted constructive control by adding voters under rule R (R-WCCAV) and weighted constructive control by deleting voters under rule R (R-WCCDV), our input contains a set of candidates C, a collection of weighted voters V (sometimes referred to as the registered voters) with preferences over C, a preferred candidate $p \in C$, and a nonnegative integer k. In R-WCCAV we also have an additional collection W of weighted voters (sometimes referred to as the unregistered voters) with preferences over C. In these problems we ask the following questions:*

1. *R-WCCAV: Is there a subcollection W' of W, of at most k voters, such that $p \in R(C, V+W')$?*

2. *R-WCCDV: Is there a subcollection V' of V, of at most k voters, such that $p \in R(C, V-V')$?*

Although in this paper we focus primarily on constructive control, Section 4.1 makes some comments about the so-called destructive variants of control problems. Given a voting rule R, weighted destructive control by adding voters under rule R (R-WDCAV) and weighted destructive control by deleting voters under rule R (R-WDCDV) are defined analogously to their constructive variants, with the only difference being that the goal is to ensure that the distinguished candidate p is not a winner. We mention in passing that throughout this paper we use what is known as the nonunique-winner model (a.k.a. the cowinner model), i.e., the goal is to make p be, or prevent p from being, an element of the winner set. We consider the nonunique-winner model to be a cleaner and more natural model than the so-called unique-winner model, in which p must be made or kept from being a one-and-only winner of the election, as that model more strongly blurs tie-breaking issues with control issues.

Note that in the above definitions the parameter k defines the number of voters that can be added/deleted, and not the total weight of the voters that can be added/deleted. This is a standard approach when modeling strategic behavior in weighted elections. For example, in the study of "R-weighted-bribery" (Faliszewski, Hemaspaandra, & Hemaspaandra, 2009), bribing each weighted voter has unit cost regardless of the voter's weight, and in the study of weighted manipulation in nearly single-peaked societies (Faliszewski, Hemaspaandra, & Hemaspaandra, 2014), the "mavericity" of a society depends on the number of so-called mavericks rather than their total weight. As to what k might be "in practice," k is reflecting the ability of the chair to add/delete voters, and so k in practice would reflect how many voters the chair is viewed as having the resources to lure in or pressure out.

We will consider approximation algorithms for WCCAV and WCCDV under t-approval and t-veto. When doing so, we will assume that input instances do not contain the integer k. Rather, the goal is simply to find (when success is possible at all) as small as possible a collection of voters to add/delete such that p is a winner of the resulting election. (Just as mentioned in the previous paragraph, we again are counting the number of added/deleted voters, not the total weight of the added/deleted voters.) For a positive integer h, an h-approximation algorithm for WCCAV/WCCDV is an algorithm that (when success is possible at all) always finds a solution that adds/deletes at most h times as many voters as an optimal action does. The notion of an $f(\cdot)$-approximation algorithm for WCCAV/WCCDV is defined analogously, where the argument to f is some variable related to the problem or instance. And the meaning of $\mathcal{O}(f(\cdot))$-approximation algorithms will be similarly clear from context. It is natural to worry about how the above seemingly incomplete definitions interact with the possibility that success might be impossible regardless of how many votes one adds/deletes. However, for t-approval WCCDV and t-veto WCCDV (and indeed, for any scoring rule), it is always possible to ensure that p is a winner, for example by deleting all the voters (recall that we are in the nonunique-winner model). For t-approval WCCAV and t-veto WCCAV, it is possible to ensure p's victory through adding voters if and only if p is a winner after we add all the unregistered voters that approve of p. These observations make it particularly easy to discuss and study approximation algorithms for t-approval and for t-veto, because we can always easily check whether there is some solution. For voting rules that don't have this easy-checking property, such an analysis might be much more complicated. The reader may wish to compare our work with Brelsford et al.'s attempt at framing a general election-problem approximation framework (Brelsford, Faliszewski, Hemaspaandra, Schnoor, & Schnoor, 2008).

In this paper we do not consider candidate-control cases (such as weighted constructive control by adding candidates and weighted constructive control by deleting candidates, WCCAC and WCCDC). The reason is that for a bounded number of candidates, when winner determination in the given weighted election system is in P it holds that both WCCAC and WCCDC are in P by brute-force search. On the other hand, if the number of candidates is not bounded then candidate control is already NP-hard for plurality (and t-approval and t-veto, in both the constructive setting and the destructive setting) even without weights (Bartholdi et al., 1992; Hemaspaandra, Hemaspaandra, & Rothe, 2007; Elkind, Faliszewski, & Slinko, 2011; Lin, 2012). Furthermore, many results for candidate control under Condorcet-consistent rules can be claimed in the weighted setting. For example, for the Maximin rule and for the Copeland family of rules, hardness results translate immediately, and it is straightforward to see that the existing polynomial-time algorithms for the unweighted cases also work for the weighted cases (Faliszewski, Hemaspaandra, & Hemaspaandra, 2011).

3.4 Weighted Coalitional Manipulation

One of our goals is to compare the complexity of weighted voter control with the complexity of weighted coalitional manipulation (WCM). WCM is similar to WCCAV in that we also add voters, but it differs in that (a) we have to add exactly a given number of voters, and (b) we can pick the preference orders of the added voters. It is quite interesting to see how the differences in these problems' definitions affect their complexities.

Definition 3.2. *Let R be a voting rule. In R-WCM we are given a weighted election (C, V), a preferred candidate $p \in C$, and a sequence $k_1, \ldots, k_n$ of positive integers. We ask whether it is possible to construct a collection $W = (w_1, \ldots, w_n)$ of n voters such that for each i, $1 \leq i \leq n$, $\omega(w_i) = k_i$, and p is a winner of the R election $(C, V{+}W)$. The voters in W are called manipulators.*

3.5 Computational Complexity

In our NP-hardness proofs we use reductions from the following NP-complete problems.

Definition 3.3. *An instance of Partition consists of a sequence $(k_1, \ldots, k_t)$ of positive integers whose sum is even. We ask whether there is a set $I \subseteq \{1, \ldots, t\}$ such that $\sum_{i \in I} k_i = \frac{1}{2}\sum_{i=1}^{t} k_i$.*

In the proof of Theorem 4.3 we will use the following restricted version of Partition, where we have greater control over the numbers involved in the problem.

Definition 3.4. *An instance of Partition$'$ consists of a sequence $(k_1, \ldots, k_t)$ of positive integers, whose sum is even, such that (a) t is an even number, and (b) for each k_i, $1 \leq i \leq t$, it holds that $k_i \geq \frac{1}{t+1}\sum_{j=1}^{t} k_j$. We ask whether there is a set $I \subseteq \{1, \ldots, t\}$ of cardinality $\frac{t}{2}$ such that $\sum_{i \in I} k_i = \frac{1}{2}\sum_{i=1}^{t} k_i$.*

Showing the NP-completeness of this problem is a standard exercise. (In particular, the NP-completeness of a variant of this problem is established as Lemma 2.3 in Faliszewski et al., 2009; the same approach can be used to show the NP-completeness of Partition$'$.) Our remaining hardness proofs are based on reductions from a restricted version of the well-known Exact-Cover-By-3-Sets problem. This restricted version is still NP-complete (Garey & Johnson, 1979).

Definition 3.5. *An instance of X3C$'$ consists of a set $B = \{b_1, \ldots, b_{3t}\}$ and a family $\mathcal{S} = \{S_1, \ldots, S_n\}$ of 3-element subsets of B such that every element of B occurs in at least one and in at most three sets in $\mathcal{S}$. We ask whether $\mathcal{S}$ contains an exact cover for B, i.e., whether there exist t sets in $\mathcal{S}$ whose union is B.*

Our choice to use X3C$'$ as the basis of some of our reductions, and the particular way we use it, will allow us to achieve something beyond simply showing that a given weighted control result is NP-complete. We will indeed be able to show that certain weighted control results for important election systems remain NP-complete even when the allowed set of weights is highly restricted, e.g., in some cases, the allowed weight set can be $\{1, 2\}$ or $\{1, 3\}$. Our cases of this sort appear within the proof of Theorem 4.13 and are highlighted in the paragraph immediately preceding that theorem.

4. Results

We now present our results. In Section 4.1 we focus on fixed numbers of candidates in scoring protocols, weakCondorcet-consistent rules, and Condorcet-consistent rules. Then in Sections 4.2 and 4.3 we consider case of an unbounded number of candidates, for t-approval and t-veto.

4.1 Bounded Numbers of Candidates

It is well-known that weighted manipulation of scoring protocols is always hard, unless the scoring protocol is in effect plurality or triviality (Hemaspaandra & Hemaspaandra, 2007). In contrast, weighted voter control is easy for m-candidate t-approval.

Theorem 4.1. *For all m and t, WCCAV and WCCDV for m-candidate t-approval are in* P*.*

Proof. Let (C, V, W, p, k) be an instance of WCCAV for m-candidate t-approval. We can assume that we add only voters who approve of p. We can also assume that we add the heaviest voters with a particular set of approvals, i.e., if we add ℓ voters approving $p, c_1, \ldots, c_{t-1}$, we can assume that we added the ℓ heaviest voters approving $p, c_1, \ldots, c_{t-1}$. Since there are only $\binom{m-1}{t-1}$—which is a constant—different sets of approvals to consider, it suffices to try all sequences of nonnegative integers $k_1, \ldots, k_{\binom{m-1}{t-1}}$ whose sum is at most k, and for each such sequence to check whether adding the heaviest k_i voters of the ith approval collection makes p a winner.

For each fixed m and t, it is clear that this algorithm, although brute-force in nature, runs in time polynomial in the input size. (The actions the algorithm uses are relatively innocuous, in fact. For example, we use sorting to group together votes within W that have identical sets of approvals, and to sort each of those in descending order of voter weight. And the number of sequences of nonnegative integers $k_1, \ldots, k_{\binom{m-1}{t-1}}$ whose sum is at most k is easily bounded by $(k+1)^{\binom{m-1}{t-1}}$, and this for fixed m and t is polynomial in the input size despite the fact that k is input in binary, because we may without loss of generality assume that $k \leq \|W\|$. We mention in passing that the $(k+1)^{\binom{m-1}{t-1}}$ bound is often wildly loose. In particular, the exact number of sequences of nonnegative integers $k_1, \ldots, k_{\binom{m-1}{t-1}}$

whose sum is exactly k' is $\binom{\binom{m-1}{t-1}+k'-1}{\binom{m-1}{t-1}-1}$. So summing that from k' equals 0 to k' equals k gives the number of sequences we will face.)

The same approach and argument work for WCCDV. Here, we delete only voters that do not approve of p, and again we delete the heaviest voters for each approval collection. Again, with m and t fixed, the running time is easily seen to be polynomial. □

One might think that the argument above works for any scoring protocol, but this is not the case. For example, consider the 3-candidate Borda instance where V consists of one weight-1 voter $b > p > a$ and W consists of a weight-2 and a weight-1 voter with preference order $a > p > b$. Then adding the weight-1 voter makes p a winner, but adding the weight-2 voter does not. And, in fact, we have the following result.[3]

Theorem 4.2. *WCCAV and WCCDV for Borda are* NP*-complete. This result holds even when restricted to a fixed number $m \geq 3$ of candidates.*

Proof. We start by considering the case of adding voters. We reduce from Partition. Given a sequence $k_1, \ldots, k_t$ of positive integers that sum to $2K$, construct an election with one registered voter of weight K voting $b > p > a > \cdots$, and t unregistered voters with weights $k_1, \ldots, k_t$ voting $a > p > b > \cdots$. Set the addition limit to t. With m candidates, the (initial) score of b is $K(m-1)$, the score of p is $K(m-2)$, and the score of a is $K(m-3)$. Thus, for p to become a winner, b's score (relative to p) needs to go down by at least K, while a's score (relative to p) should not go up by more than K. It follows that $k_1, \ldots, k_t$ has a partition if and only if p can be made a winner.

We use the same construction for the deleting voters case. Now, all voters are registered and the deletion limit is t. Since we can't delete all voters and since our goal is to make p a winner, we cannot delete the one voter voting $b > p > a > \cdots$ (since then a would be the unique winner). The rest of the argument is identical to the case of adding voters. □

Interestingly, it is possible to extend the above proof to work for all scoring protocols other than t-approval (the main idea stays the same, but the technical details are more involved). And so, regarding the complexity of WCCAV and WCCDV for scoring protocols with a fixed number of candidates, the cases of Theorem 4.1 are the only P cases (assuming P $\neq$ NP).

Theorem 4.3. *For each scoring protocol $(\alpha_1, \ldots, \alpha_m)$, if there exists an i, $1 < i < m$, such that $\alpha_1 > \alpha_i > \alpha_m$, then WCCAV and WCCDV for $(\alpha_1, \ldots, \alpha_m)$ are* NP*-complete.*

Proof. Let $\alpha = (\alpha_1, \ldots, \alpha_m)$ be a scoring protocol such that there is an i such that $\alpha_1 > \alpha_i > \alpha_m$. Let δ be the third largest value in the set $\{\alpha_1, \ldots, \alpha_m\}$. We will show that WCCAV and WCCDV are NP-complete for scoring protocol $\beta = (\beta_1, \ldots, \beta_m) = (\alpha_1 - \delta, \ldots, \alpha_m - \delta)$. While formally we have defined scoring protocols to contain only nonnegative values, using β simplifies our construction and does not affect the correctness of the proof. To further simplify notation, given some candidates $x_1, \ldots, x_\ell$, by $F[x_1 = \beta_{i_1}, x_2 = \beta_{i_2}, \ldots, x_\ell = \beta_{i_\ell}]$ we mean a fixed preference order that ensures, under β, that each x_j, $1 \leq j \leq \ell$, is ranked

3. An analogue of this theorem in the model in which we are bounding the total weight of votes that can be added/deleted was obtained by Russell (2007).

at a position that gives β_{i_j} points. (The candidates not mentioned in the $F[\dots]$ notation are ranked arbitrarily.) We let γ_1, γ_2, and γ_3 be the three highest values in the set $\{\beta_1, \dots, \beta_m\}$. Clearly, $\beta_1 = \gamma_1 > \gamma_2 > \gamma_3 = 0$. (Note that γ_2 might be different from β_2, and γ_3 might be different from β_3. For example, if $\beta = (3, 3, 2, 0, 0, -1, -1)$, then $\gamma_1 = 3$, $\gamma_2 = 2$, and $\gamma_3 = 0$, but $\beta_1 = 3$, $\beta_2 = 3$, and $\beta_3 = 2$.)

We give a reduction from Partition to β-WCCAV (the membership of β-WCCAV in NP is clear); let $(k_1, \dots, k_t)$ be an instance of Partition, i.e., a sequence of positive integers that sum to $2K$. We form an election $E = (C, V)$ where $C = \{p, a, b, c_4, \dots, c_m\}$ and where the collection V contains the following three groups of voters (for the WCCAV part of the proof below, we set $T = 1$; for the WCCDV part of the proof we will use the same construction but with a larger value of T):

1. A group of T voters, each with weight K and preference order $F[b = \gamma_1, a = \gamma_2, p = 0]$.

2. A group of T voters, each with weight K and preference order $F[p = \gamma_1, b = \gamma_2, a = 0]$.

3. For each $c_i \in C$, there are 6 collections of $2T$ voters, one collection for each permutation (x, y, z) of (p, a, b); the voters in each collection have weight K and preference order $F[x = \beta_1, y = \beta_2, z = \beta_3, c_i = \beta_m]$.

Let M be the number of points that each of a, b, and p receive from the third group of voters (each of these candidates receives the same number of points from these voters). For each $c_i \in C$ and each $x \in \{p, a, b\}$, x receives at least $4TK\gamma_1$ points more than c_i from the voters in the third group (in each vote in the third group, x receives at least as many points as c_i, and there are two collections of $2T$ voters where x receives $\beta_1 = \gamma_1$ points and c_i receives $\beta_m \leq 0$ points). Thus it holds that our candidates have the following scores:

1. p has $M + TK\gamma_1$ points,

2. a has $M + TK\gamma_2$ points,

3. b has $M + TK(\gamma_1 + \gamma_2)$ points, and

4. each candidate $c_i \in C$ has at most $M - 2TK\gamma_1$ points (each $c_i \in C$ receives at most $M - 4TK\gamma_1$ points from the third group of voters and at most $2TK\gamma_1$ points from the first two groups of voters).

As a result, b is the unique winner. There are t unregistered voters with weights $Tk_1, \dots, Tk_t$, each with preference order $F = [a = \gamma_1, p = \gamma_2, b = 0]$. We set the addition limit to be t. It is clear that irrespective of which voters are added, none of the candidates in $\{c_4, \dots, c_m\}$ becomes a winner.

If there is a subcollection of $(k_1, \dots, k_t)$ that sums to K, then adding corresponding unregistered voters to the election ensures that all three of p, a, and b are winners (each with score $M + TK(\gamma_1 + \gamma_2)$). On the other hand, assume that there are unregistered voters of total weight TL, whose addition to the election ensures that p is among the winners. After adding these voters to the election, p has $M + TK\gamma_1 + TL\gamma_2$ points, a has $M + TL\gamma_1 + TK\gamma_2$ points, and b has $M + TK\gamma_1 + TK\gamma_2$ points. For p to have score at least as high as b, we must have that $L \geq K$. However, for a not to have score higher than p, it must be the case that $L \leq K$ (recall that $\gamma_1 > \gamma_2$). This means that $L = K$. Thus it is possible to ensure that p is a winner of the election by adding at most t unregistered voters if and only if there is a subcollection of $(k_1, \ldots, k_t)$ that sums to K. And, completing the proof, we note that the reduction can be carried out in polynomial time.

Let us now move on to the case of WCCDV. We will use the same construction, but with the following modifications:

1. Our reduction is now from Partition$'$. Thus without loss of generality we can assume that t is an even number and that for each i, $1 \leq i \leq t$, it holds that $k_i \geq \frac{1}{1+t}2K$.

2. We set $T = \left\lceil \frac{t}{2}(t+1)\frac{\gamma_1}{\gamma_1 - \gamma_2} \right\rceil + 1$ (the reasons for this choice of T will become apparent in the course of the proof; intuitively it is convenient to think of T as of a large value that, nonetheless, is polynomially bounded with respect to t).

3. We include the unregistered voters as "the fourth group of voters."

4. We set the deletion limit to $\frac{t}{2}$.

Including the fourth group of voters, candidates have the following scores: p has $M + TK\gamma_1 + 2TK\gamma_2$ points, a has $M + TK\gamma_2 + 2TK\gamma_1$ points, b has $M + TK(\gamma_1 + \gamma_2)$ points, and each candidate $c_i \in C$ has at most M points.

By the same reasoning as in the WCCAV case, we see that if there is a size-$\frac{t}{2}$ subcollection of $k_1, \ldots, k_t$ that sums to K, then deleting the corresponding voters ensures that p is among the winners (together with a and b); we may imagine that we first remove all the voters from the fourth group and then add back those $\frac{t}{2}$ of them, whose weights sum to TK. We now show that if there is a way to delete up to $\frac{t}{2}$ voters to ensure that p is among the winners, then the deleted voters must come from the fourth group, must have total weight K, and there must be exactly $\frac{t}{2}$ of them. For the sake of contradiction, let us assume that it is possible to ensure p's victory by deleting up to $\frac{t}{2}$ voters, of whom fewer than $\frac{t}{2}$ come from the fourth group. Let s be the number of deleted voters from the fourth group ($s < \frac{t}{2}$) and let x be a real number such that xTK is their total weight. We have that xTK is at most (see below for explanation regarding the first inequality)

$$xTK \leq 2TK - \frac{t-s}{1+t}(2TK) \leq 2TK\left(1 - \frac{\frac{t}{2}+1}{1+t}\right) = TK\frac{t}{1+t}.$$

That is, we have $0 \leq x \leq \frac{t}{1+t}$. To see why the first inequality holds, recall that the lowest weight of a voter from the fourth group is at least $\frac{1}{1+t}2TK$ (because we reduce from Partition$'$). Thus the highest total weight of s voters from the fourth group is, at most, the total weight of the fourth-group voters ($2TK$) less the weight of the lightest $t - s$ voters from this group (which is at least $\frac{t-s}{1+t}(2TK)$).

Prior to deleting any voters, a has $TK(\gamma_1 - \gamma_2)$ points more than p. After deleting the s voters from the fourth group, this difference decreases to $TK(1-x)(\gamma_1 - \gamma_2)$. If we additionally delete up to $\frac{t}{2}$ voters from the first three groups of voters, each with weight K, then the difference between the scores of a and p decreases, at most, to the following value (note that in each deleted vote both a and p are ranked at positions where they receive γ_1,

γ_2, or 0 points):

$$TK(1-x)(\gamma_1-\gamma_2)-\frac{t}{2}K\gamma_1 \geq TK\frac{1}{t+1}(\gamma_1-\gamma_2)-\frac{t}{2}K\gamma_1 = K\left(\frac{T(\gamma_1-\gamma_2)}{t+1}-\frac{\frac{t}{2}(t+1)\gamma_1}{t+1}\right) > 0.$$

The final inequality follows by our choice of T. The above calculation shows that if there is a way to ensure p's victory by deleting up to $\frac{t}{2}$ voters then it requires deleting exactly $\frac{t}{2}$ voters from the fourth group. The same reasoning as in the case of WCCAV shows that these $\frac{t}{2}$ deleted voters must correspond to a size-$\frac{t}{2}$ subcollection of $(k_1, \ldots, k_t)$ that sums to K. ☐

As a side comment, we mention that WDCAV and WDCDV for scoring protocols (that is, the destructive variants of WCCAV and WCCDV) have simple polynomial-time algorithms: It suffices to loop through all candidates c, $c \neq p$, and greedily add/delete voters to boost the score of c relative to p as much as possible.

Theorem 4.4. *For each scoring protocol $\alpha = (\alpha_1, \ldots, \alpha_m)$, α-WDCAV and α-WDCDV are in* P.

Combining Theorems 4.1 and 4.3, we obtain the following corollary, which we contrast with an analogous result for WCM (Hemaspaandra & Hemaspaandra, 2007); we also mention in passing the recent attainment of a dichotomy result for voter control under so-called pure scoring rules, for *unweighted* elections and an unbounded number of candidates (Hemaspaandra, Hemaspaandra, & Schnoor, 2014).

Corollary 4.5. *For each scoring protocol $(\alpha_1, \ldots, \alpha_m)$ the problems WCCAV and WCCDV are* NP*-complete if $\|\{\alpha_1, \ldots, \alpha_m\}\| \geq 3$ and are in* P *otherwise.*

Theorem 4.6 (Hemaspaandra & Hemaspaandra, 2007)**.** *For each scoring protocol $(\alpha_1, \ldots, \alpha_m)$, $m \geq 2$, WCM is* NP*-complete if $\alpha_2 > \alpha_m$ and is in* P *otherwise.*

We see that for scoring protocols with a fixed number m of candidates, either WCM is harder than WCCAV and WCCDV (for the case of t-approval with $2 \leq t < m$), or the complexity of WCM, WCCAV, and WCCDV is the same (P-membership for plurality and triviality, and NP-completeness for the remaining cases). One may wonder which property of WCM is responsible for the fact that for t-approval, $2 \leq t \leq m$, WCM is harder than WCCAV and WCCDC. Speaking very informally, the answer is that WCM intimately involves the instantiation of the values of the (initially unspecified) votes of the manipulators, and in this particular setting that is, in effect, requiring them to solve a Partition problem. On the other hand, in WCCAV and WCCDV the preference orders of all the voters are fixed in the input, and the chair chooses only which votes to add; this, for example, facilitated the polynomial-time algorithm in the proof of Theorem 4.1.

There are, nonetheless, voting rules for which WCM is easier than WCCAV and WCCDV. This happens, for example, if on the one hand in WCCAV and WCCDV under the particular rule the chair has to balance out differing votes in a way that makes these problems hard, yet on the other hand for WCM under the particular rule we can show that if there is any successful manipulation then there is one in which all the manipulators cast identical votes. Theorem 4.7, Corollary 4.8, and their proofs present exactly such a case.

Theorem 4.7. *For every weakCondorcet-consistent election system and for every Condorcet-consistent election system, WCCAV and WCCDV are* NP*-hard. This result holds even when restricted to a fixed number $m \geq 3$ of candidates.*

Proof. To show that WCCAV is NP-hard, we reduce from Partition. Given a sequence $k_1, \ldots, k_t$ of positive integers that sum to $2K$, construct an election with two registered voters, one voter with weight 1 voting $p > a > b > \cdots$ and one voter with weight $2K$ voting $b > p > a > \cdots$, and t unregistered voters with weights $2k_1, \ldots, 2k_t$ voting $a > p > b > \cdots$. Set the addition limit to t. Suppose we add unregistered voters to the election with a total vote weight equal to $2L$.

- If $L < K$, then b is the Condorcet winner, and thus the unique winner of the election.
- If $L > K$, then a is the Condorcet winner, and thus the unique winner of the election.
- If $L = K$, then p is the Condorcet winner, and thus the unique winner of the election.

The WCCDV case uses the same construction. Now, all voters are registered and the deletion limit is t. Since we can delete at most t of our $t + 2$ voters, and since our goal is to make p a winner, we can't delete the sole voter voting $b > p > a$, since then a would be the Condorcet winner. The rest of the argument is similar to the adding voters case. □

Recall from Section 3 that Condorcet denotes the election system whose winner set is exactly the set of Condorcet winners, and weakCondorcet denotes the election system whose winner set is exactly the set of weak Condorcet winners.

Corollary 4.8. *For Condorcet and weakCondorcet, WCM is in* P *and WCCAV and WCCDV are* NP*-complete. This result holds even when restricted to a fixed number $m \geq 3$ of candidates.*

Proof. It is immediate that WCM for Condorcet and weakCondorcet are in P. To see if we have a "yes"-instance of WCM, it suffices to check whether letting all the manipulators rank p (the preferred candidate) first and ranking all the remaining candidates in some arbitrary order ensures p's victory. NP-completeness of WCCAV and WCCDV follows directly from Theorem 4.7. □

Condorcet and weakCondorcet do not always have winners. For those who prefer their voting systems to always have at least one winner, we note that WCM for 3-candidate Llull is in P (Faliszewski et al., 2008).

Corollary 4.9. *For 3-candidate Llull, WCM is in* P *and WCCAV and WCCDV are* NP*-complete.*

The main results of this section are also presented in Table 1 of Section 6.

4.2 t-Approval and t-Veto with an Unbounded Number of Candidates

Let us now look at the cases of t-approval and t-veto rules, for an unbounded number of candidates. The reason we focus on these is that these are the most interesting families of scoring protocols whose complexity has not already been resolved in the previous section. The reason we say that is that Theorem 4.3 shows that whenever we have at least three distinct values in a scoring vector, we have NP-completeness. So any scoring-protocol family that, for some number of candidates, has three distinct values in its scoring vector is NP-hard for WCCAV and WCCDV. Thus the really interesting cases are indeed t-approval and t-veto.

Our starting point here is the work of Lin (2012), which showed that for $t \geq 4$, WCCAV for t-approval and WCCDV for t-veto are NP-complete, and that for $t \geq 3$, WCCDV for t-approval and WCCAV for t-veto are NP-complete. These results hold even for the unweighted case. It is also known that the remaining unweighted cases are in P (Bartholdi et al., 1992; Lin, 2012) and that WCCAV and WCCDV for plurality and veto are in P (Lin, 2012). In this section, we look at and solve the remaining open cases, WCCAV for 2-approval, 3-approval, and 2-veto, and WCCDV for 2-approval, 2-veto, and 3-veto. We start by showing that 2-approval-WCCAV is in P. We point out that our proof techniques (especially the polynomial-time algorithms for 2-approval-WCCAV and for 2-veto-WCCDV) are quite different from those of Lin (2012).

Theorem 4.10. *WCCAV for 2-approval is in* P.

Proof. We claim that Algorithm 1 solves 2-approval-WCCAV in polynomial time. (In this algorithm and the proof of correctness, whenever we speak of the r heaviest voters in voter set X, we mean the $\min(r, \|X\|)$ heaviest voters in X.)

We note that we will add only voters that approve of p. Thus we delete from W all the voters who do not approve of p.

Let us consider the repeat-until loop in Algorithm 1. If we reject in the first iteration of this loop (in the first forall loop) then, clearly, there is no solution for the given instance. Furthermore, we claim that if there is a solution to the input instance, then after the second forall loop it is still possible to find it. To see this, consider some candidate $c \in C - \{p\}$ and some number $\ell \in \{1, \dots, k-1\}$. If the sum of the weights of $k - \ell$ heaviest voters in W that do not approve of c is less than s_c (that is, is less than the difference between the score of c and the score of p in the original election), then we certainly need to add at least $k - \ell + 1$ voters who do not approve of c. However, since altogether we can add at most k voters, this means that we can add at most $\ell - 1$ voters who do approve of c. In effect, we can safely delete from W all but $\ell - 1$ heaviest voters who approve of c (as in the proof of Theorem 4.1, if we decide to add some r voters approving $\{p, c\}$, we may assume that we add the r heaviest voters approving $\{p, c\}$; thus keeping only the $\ell - 1$ heaviest voters that approve of $\{p, c\}$ is a correct strategy).

So, if we reject in the first iteration of the repeat-until loop, then certainly there is no solution for the input instance, and if we do not, then we start the second iteration with an instance that has a solution if and only if the original one had. Thus, by induction, we never reject incorrectly in the repeat-until loop. If we get through the repeat-until without rejecting, and we have fewer than k voters left in W, then adding all of W is the best we can do (since all voters in W approve p).

Algorithm 1: 2-approval-WCCAV

Input: (C, V, W, p, k)

forall $c \in C - \{p\}$ **do**
  let $s_c = score_{(C,V)}(c) - score_{(C,V)}(p)$.
Delete from W all voters that do not approve of p.
repeat
  forall $c \in C - \{p\}$ **do**
    if *the sum of the weights of the k heaviest voters in W that do not approve of c is less than s_c* **then reject**
    // It is impossible to get $score(c) \le score(p)$ by adding less than or equal to k voters from W.
  forall $c \in C - \{p\}$ **and** $\ell \in \{1, \ldots, k-1\}$ **do**
    if *the sum of the weights of the $k - \ell$ heaviest voters in W that do not approve of c is less than s_c* **then**
      delete from W all voters approving c except for the $\ell - 1$ heaviest such voters.
      // We need to add at least $k - \ell + 1$ voters that do not approve of c, and so we can add at most $\ell - 1$ voters approving c.
until *no more changes.*
if $\|W\| \ge k$ **then accept** *// We can make p a winner by adding the k heaviest voters from W.*
if $\|W\| < k$ **then**
  if *adding all of W will make p a winner* **then accept else reject**

On the other hand, if we get through the repeat-until loop, and we have at least k voters left in W, then adding the k heaviest voters from W will make p a winner. Why? Let c be a candidate in $C-\{p\}$. Let r be the number of voters from W that are added and that approve of c. Since we made it through the repeat-until, we know that [the sum of the weights of the k heaviest voters in W that do not approve of c] is at least s_c (because we did not reject in the first forall loop). We will show that after adding the voters, $score(c) - score(p) \le 0$, which implies that p is a winner. If $r = 0$, $score(c) - score(p) = s_c$ - [the sum of the weights of the k heaviest voters in W] ≤ 0. If $r > 0$, then [the sum of the weights of the $k - r$ heaviest voters in W that do not approve of c] is at least s_c (for otherwise we would have at most $r - 1$ voters approving c left in W due to the if statement in the second forall loop). And so $score(c) - score(p) = s_c$ - [the sum of the weights of the $k - r$ heaviest voters in W that do not approve of c] ≤ 0. ☐

Theorem 4.11. *WCCDV for 2-veto is in* P*.*

Instead of proving this theorem directly, we show a more general relation between the complexity of t-approval/t-veto WCCAV and WCCDV.

Theorem 4.12. *For each fixed t, it holds that t-veto-WCCDV (t-approval-WCCDV) polynomial-time many-one reduces to t-approval-WCCAV (t-veto-WCCAV).*

Proof. We first give a reduction from t-veto-WCCDV to t-approval-WCCAV. The idea is that deleting a t-veto vote v from t-veto election (C, V) is equivalent, in terms of net effect on the scores, to adding a t-approval vote v' to this election, where v' approves exactly of the t candidates that v disapproves of. The problem with this approach is that we are to reduce t-veto-WCCDV to t-*approval*-WCCAV and thus we have to show how to implement t-veto scores with t-approval votes.

Let (C, V, p, k) be an instance of t-veto-WCCDV, where $V = (v_1, \dots, v_n)$. Let $m = \|C\|$. Let $\omega_{\max}$ be the highest weight of a vote in V. We set D to be a set of up to $t-1$ new candidates, such that $\|C\| + \|D\|$ is a multiple of t. We set V_0 to be a collection of $\frac{\|C\|+\|D\|}{t}$ t-approval votes, where each vote has weight $\omega_{\max}$ and each candidate in $C \cup D$ is approved in exactly one of the votes. For each vote v_i in V we create a set $C_i = \{c_i^1, \dots, c_i^{(t-1)(m-t)}\}$ of candidates and we create a collection of voters $V_i = (v_i^1, \dots, v_i^{m-t})$. Each voter v_i^j, $1 \leq j \leq m-t$, has weight $\omega(v_i)$ and approves of the jth candidate approved by v and of the $t-1$ candidates $c_i^{(j-1)(t-1)+1}, \dots, c_i^{j(t-1)}$.

We form an election $E' = (C', V')$, where $C' = C \cup D \cup \bigcup_{i=1}^{n} C_i$ and $V' = V_0 + V_1 + \cdots + V_n$. For each candidate c, let s_c be c's t-veto score in (C, V); we see that c's t-approval score in E' is $\omega_{\max} + s_c$ (every candidate from C receives a single approval from one weight $\omega_{\max}$ voter from V_0 and for each voter v_i in V and for each candidate c that v_i approves of, there is a unique voter in V_i that has the same weight as v_i and that approves of c). Furthermore, each candidate $c \in C' - C$ has t-approval score at most $\omega_{\max}$ in E' (each candidate in $C' - C$ is approved by exactly one voter in V' and each voter in V' has weight at most $\omega_{\max}$).

We form an instance (C', V', W, p, k) of t-approval-WCCAV, where $W = (w_1, \dots, w_n)$, and for each i, $1 \leq i \leq n$, $\omega(w_i) = \omega(v_i)$, and w_i approves exactly of those candidates that v_i disapproves of; adding voter w_i to t-approval election (C', V') has the same net effect on the scores of the candidates in C as does deleting v_i from t-veto election (C, V). (The role of the candidates in D is to pad the election so that it is easy to use t-approval votes—those in V_0—to ensure that the candidates in C have at least as many points as the other candidates, irrespective of which voters we add.) This completes the reduction.

Let us now give a reduction from t-approval-WCCDV to t-veto-WCCAV. The idea is the same as in the previous reduction and the main difficulty of the proof is to show how to implement t-approval scores with t-veto votes.[4] In particular, the role of the candidates in D is, again, to provide a convenient way of padding the election and implementing the scores of the other candidates. However, this time the construction is more involved because of the nature of t-veto: as opposed to the case of t-approval, under t-veto if we add a candidate to the election then the total number of candidates approved per vote increases.

Let (C, V, p, k) be an instance of t-approval-WCCDV, where $V = (v_1, \dots, v_n)$. Let $m = \|C\|$ and let $\omega_{\max}$ be the highest weight of a vote in V. We set D to be a set of candidates such that $t \leq \|D\| \leq 2t-1$ and $\|C\| + \|D\| = s \cdot t$ for some integer s, $s \geq 3$ (note that for our setting to not be trivial it must be the case that $m > t$). We set V_0 to be a collection of $4n(s-2)$ (t-veto) votes (over candidate set $C \cup D$), each with weight $\omega_{\max}$; each candidate from C is approved in all these votes whereas each candidate from D is disapproved in at least half of them (since $t \leq \|D\| \leq 2t-1$, it is easy to construct

4. The reader may wonder why we do not simply use the previous argument by applying it to $(m-t)$-veto and $(m-t)$-approval. The reason is that given an instance of $(m-t)$-veto-WCCDV (with m candidates and n voters), our reduction would output an instance of $(m-t)$-approval-WCCAV with more than m candidates. Thus it would not be correct to interpret this instance as a t-veto-WCCAV instance.

such votes[5]). For each vote v_i in V, we create a collection V_i of $(s-1)$ votes satisfying the following requirements: (a) each candidate approved in v_i is also approved in each of the votes in V_i, and (b) each candidate not approved in v_i, is approved in exactly $(s-2)$ votes in V_i. (Such votes are easy to construct: We always place the top t candidates from v_i in the top t positions of the vote; for the remaining positions, in the first vote we place the candidates in some arbitrary, easily computable order, and in each following vote we shift these candidates cyclically by t positions with respect to the previous vote.) Each vote in V_i has weight $\omega(v_i)$.

We form an election $E' = (C', V')$, where $C' = C \cup D$ and $V' = V_0 + V_1 + \cdots + V_n$. For each candidate c, let s_c be c's t-approval score in (C, V); we see that c's t-veto score in E' is $4n(s-2)\omega_{\max} + (s-2)(\sum_{i=1}^{n} \omega(v_i)) + s_c$ (c is approved by every voter from V_0 and by at least $s-2$ voters from each group V_i, $1 \leq i \leq n$; additionally, for every voter v_i that approves of c, there is the $(s-1)$'th voter in group V_i that approves of c). Furthermore, each candidate from D has t-veto score at most $3n(s-2)\omega_{\max}$ in E' (each of them gets at most $2n(s-2)\omega_{\max}$ points from the voters in V_0 and at most $(s-2)\omega_{\max}$ points from each V_i, $1 \leq i \leq n$).

We form an instance (C', V', W, p, k) of t-veto-WCCAV, where $W = (w_1, \ldots, w_n)$, and for each i, $1 \leq i \leq n$, $\omega(w_i) = \omega(v_i)$, and w_i disapproves of exactly those candidates that v_i approves of; adding voter w_i to t-veto election (C', V') has the same net effect on the scores of candidates in C as deleting voter v_i from t-approval election (C, V) has. Furthermore, since each candidate in D has at least $n\omega_{\max}$ fewer points than each candidate in C, the fact that adding w_i increases scores of candidates in D does not affect the correctness of our reduction. $\square$

All other remaining cases (WCCDV for 2-approval, WCCAV for 3-approval, WCCAV for 2-veto, and WCCDV for 3-veto) are NP-complete. Interestingly, in contrast to many other NP-complete weighted election problems, we need only a very limited set of weights to make the reductions work. Namely, due to the choice of reducing from X3C′ and due to the particular reductions we build, the proof of the following theorem establishes (the details of why are given within the proof) that (a) for every pair of integers $1 \leq a < b$, it holds that WCCDV for 2-approval and WCCAV for 2-veto are NP-complete even when the legal set of weights is restricted to be $\{a, b\}$, and (b) WCCDV for 3-approval and WCCAV for 3-veto are NP-complete even when the legal set of weights is restricted to be $\{1, 3\}$.

Theorem 4.13. *WCCAV for 2-veto and 3-approval and WCCDV for 2-approval and 3-veto are* NP*-complete.*

Proof. Membership in NP is immediate, so it suffices to prove NP-hardness. We will first give the proof for WCCDV for 2-approval. By Theorem 4.12 this also immediately gives the result for WCCAV for 2-veto. We will reduce from X3C′ from Definition 3.5. Let $B = \{b_1, ..., b_{3t}\}$ and let $\mathcal{S} = \{S_1, ..., S_n\}$ be a family of 3-element subsets of B such that every element of B occurs in at least one and in at most three sets in $\mathcal{S}$. We construct the

5. Here is one possible construction. Let $D = \{d_1, \ldots, d_\ell\}$, where $t \leq \ell \leq 2t-1$. We form sets $D_0 = \{d_1, \ldots, d_t\}$ and $D_1 = \{d_\ell, \ldots, d_{\ell-t+1}\}$. We have that $D = D_0 \cup D_1$ (D_0 and D_1 might overlap). There are an even number of voters in V_0; exactly half of them disapprove of the candidates from the set D_0 and exactly half disapprove of the candidates from the set D_1.

following instance (C, V, p, k) of WCCDV for 2-approval. We set $C = \{p\} \cup \{b_j \mid 1 \le j \le 3t\} \cup \{s_i, s'_i \mid 1 \le i \le n\} \cup \{d_0, d_1, \ldots, d_{3t}\}$ ($d_0, d_1, \ldots, d_{3t}$ are dummy candidates that are used for padding). For $1 \le j \le 3t$, let ℓ_j be the number of sets in $\mathcal{S}$ that contain b_j. By assumption, for each j, $1 \le j \le 3t$, we have that $1 \le \ell_j \le 3$. V consists of the following voters:

weight	preference order	
2	$s_i > s'_i > \cdots$	for all $1 \le i \le n$ and $S_i = \{b_{i_1}, b_{i_2}, b_{i_3}\}$
1	$s_i > b_{i_1} > \cdots$	
1	$s_i > b_{i_2} > \cdots$	
1	$s'_i > b_{i_3} > \cdots$	
2	$p > d_0 > \cdots$	
$3 - \ell_j$	$b_j > d_j > \cdots$	for all $1 \le j \le 3t$ such that $\ell_j < 3$.

Note that $score(s_i) = 4$, $score(s'_i) = 3$, $score(b_j) = 3$, $score(p) = 2$, and $score(d_j) \le 2$. We set $k = n + 2t$ and we claim that $\mathcal{S}$ contains an exact cover if and only if p can become a winner after deleting at most $n + 2t$ voters.

($\Rightarrow$): Delete the $(n - t)$ weight-2 voters corresponding to the sets not in the cover and delete the $3t$ weight-1 voters corresponding to the sets in the cover. Then the score of p does not change, the score of each s_i decreases by 2, the score of each s'_i decreases by at least 1, and the score of each b_j decreases by 1. So, p is a winner.

($\Leftarrow$): We need to delete $3t$ voters to decrease the score of every b_j candidate by 1. (Note that there is no reason to delete the voters with preference orders of the form $b_j > d_j > \cdots$ ($1 \le j \le 3t$). It suffices to decrease the score of each b_j by one and, since we also need to decrease the scores of candidates s_i and s'_i ($1 \le i \le n$), it is always better to delete voters with preference orders of the form $s_i > b_j > \cdots$ and $s'_i > b_j > \cdots$.) After deleting these $3t$ voters, there are at most t values of i, $1 \le i \le n$, such that the score of s_i and the score of s'_i are at most 2 (for each i, obtaining the score at most 2 for candidates s_i and s'_i takes at least 3 unique voters of the $3t$ deleted ones).

If there are exactly t values of i, $1 \le i \le n$, such that the score of s_i and the score of s'_i are at most 2, then these t values of i correspond to a cover. (Why is this so? We consider a situation where we have already deleted the $3t$ voters with preference orders of the forms $s_i > b_j > \cdots$ and $s'_i > b_j > \cdots$, where $b_j \in S_i$. If, after deleting these voters, for some i the scores of both s_i and s'_i decreased to 2, we must have deleted exactly the three voters that correspond to members of S_i. Thus, if after deleting voters corresponding to $3t$ members of B we ensured that there are t values i such that the scores of s_i and s'_i decreased to 2, then it must be the case that these values of i correspond to a cover.) If there are less than t values of i, $1 \le i \le n$, such that the score of s_i and the score of s'_i are at most 2, then the remaining voters that are deleted, and there are at most $n - t$ of them, need to decrease the score of s_i and/or s'_i for more than $n - t$ values of i, $1 \le i \le n$. But that is not possible, since there is no voter that approves of both s_i or s'_i and s_j or s'_j for $i \ne j$.

Note that this construction uses only weights 1 and 2. In fact, we can establish NP-completeness for WCCDV for 2-approval for every set of allowed weights of size at least two (note that if the set of weights has size one, the problem is in P, since this is in essence the unweighted case resolved by Lin, 2012). Since the reductions of Theorem 4.12 do not change the set of voter weights, we have the same result for WCCAV for 2-veto.

So, suppose our weight set contains w_1 and w_2, $w_2 > w_1 > 0$. We modify the construction above as follows. We keep the same set of candidates and we change the voters as follows.

$$
\begin{array}{ccl}
\# & \text{weight} & \text{preference order}
\end{array}
$$
$$
\left.\begin{array}{ccl}
1 & w_2 & s_i > s'_i > \cdots\\
1 & w_1 & s_i > b_{i_1} > \cdots\\
1 & w_1 & s_i > b_{i_2} > \cdots\\
1 & w_1 & s'_i > b_{i_3} > \cdots
\end{array}\right\} \text{for all } 1 \leq i \leq n \text{ and } S_i = \{b_{i_1}, b_{i_2}, b_{i_3}\}
$$
$$
\begin{array}{ccll}
2 & w_1 & p > d_0 > \cdots & \text{if } w_2 \leq 2w_1\\
1 & w_2 & p > d_0 > \cdots & \text{if } w_2 > 2w_1\\
\ell - \ell_j & w_1 & b_j > d_j > \cdots & \text{for all } 1 \leq j \leq 3t.
\end{array}
$$

Here, ℓ is the smallest integer such that $\ell w_1 > \max(2w_1, w_2)$. Note that $\ell \geq 3$ and so $\ell - \ell_j$ is never negative. Note that $\mathit{score}(s_i) = w_2 + 2w_1$, $\mathit{score}(s'_i) = w_2 + w_1$, $\mathit{score}(b_j) = \ell w_1$, $\mathit{score}(p) = \max(2w_1, w_2)$, and $\mathit{score}(d_j) \leq \max(2w_1, w_2)$. The same argument as above shows that $\mathcal{S}$ contains an exact cover if and only if p can become a winner after deleting at most $n + 2t$ voters.

We now turn to the proof for WCCDV for 3-veto. Our construction will use only weights 1 and 3. Since the reductions of Theorem 4.12 do not change the set of voter weights, weights 1 and 3 also suffice to get NP-completeness for WCCAV for 3-approval. Given the instance of X3C$'$ described above, we construct the following instance (C, V, p, k) of WCCDV for 3-veto. We set $C = \{p\} \cup B \cup \{s_i \mid 1 \leq i \leq n\} \cup \{r, d, d'\}$ (d and d' are dummy candidates that are used for padding) and V consists of the following voters:

$$
\begin{array}{ccl}
\# & \text{weight} & \text{preference order}
\end{array}
$$
$$
\left.\begin{array}{ccl}
1 & 3 & \cdots > p > s_i > r\\
1 & 1 & \cdots > p > s_i > b_{i_1}\\
1 & 1 & \cdots > p > s_i > b_{i_2}\\
1 & 1 & \cdots > p > s_i > b_{i_3}
\end{array}\right\} \text{for all } 1 \leq i \leq n \text{ and } S_i = \{b_{i_1}, b_{i_2}, b_{i_3}\}
$$
$$
\begin{array}{ccll}
3n - 3t & 1 & \cdots > d > d' > r & \\
3n - 3 & 1 & \cdots > d > d' > s_i & \text{for all } 1 \leq i \leq n\\
3n + 1 - \ell_j & 1 & \cdots > d > d' > b_j & \text{for all } 1 \leq j \leq 3t.
\end{array}
$$

It is more convenient to count the number of vetoes for each candidate than to count the number of approvals. Note that $\mathit{vetoes}(s_i) = 3n+3$, $\mathit{vetoes}(b_j) = 3n+1$, $\mathit{vetoes}(r) = 6n-3t$, $\mathit{vetoes}(p) = 6n$, and $\mathit{vetoes}(d) = \mathit{vetoes}(d') \geq 3n$. We claim that $\mathcal{S}$ contains an exact cover if and only if p can become a winner (i.e., have a lowest number of vetoes) after deleting at most $n + 2t$ voters.

($\Rightarrow$): Delete the $(n - t)$ weight-3 voters corresponding to the sets not in the cover and delete the $3t$ weight-1 voters that veto p and that correspond to the sets in the cover. Then $\mathit{vetoes}(s_i) = \mathit{vetoes}(b_j) = \mathit{vetoes}(r) = \mathit{vetoes}(p) = 3n$ and $\mathit{vetoes}(d) = \mathit{vetoes}(d') \geq 3n$. So, p is a winner.

($\Leftarrow$): We can assume that we delete only voters that veto p. Suppose we delete k_1 weight-1 voters and k_2 weight-3 voters, $k_1 + k_2 \leq n + 2t$. After this deletion, $\mathit{vetoes}(p) = 6n - k_1 - 3k_2$, $\mathit{vetoes}(r) = 6n - 3t - 3k_2$, and $\mathit{vetoes}(b_j) \leq 3n + 1$. In order for p to be a winner, we need $\mathit{vetoes}(p) \leq \mathit{vetoes}(r)$. This implies that $k_1 \geq 3t$. We also need $\mathit{vetoes}(p) - \mathit{vetoes}(b_j) \leq 0$.

Since $vetoes(p) - vetoes(b_j) \geq 6n - k_1 - 3k_2 - (3n+1) \geq 6n - (n+2t-k_2) - 3k_2 - 3n - 1 = 2n-2t-2k_2-1$, it follows that $k_2 \geq n-t$. (To see that this is the case, note that if we require that $vetoes(p) - vetoes(b_j) \leq 0$ and we know that $vetoes(p) - vetoes(b_j) \geq 2n - 2t - 2k_2 - 1$, then we must require that $2n-2t-2k_2-1 \leq 0$. This expression is equivalent to $k_2 \geq n-t-\frac{1}{2}$. Since k_2, n, and t are integers, it must be the case that $k_2 \geq n-t$.) So we delete $3t$ weight-1 votes and $n-t$ weight-3 votes, and after deleting these voters $vetoes(p) = 3n$. In order for p to be a winner, we can delete at most one veto for each b_j and at most three vetoes for each s_i. This implies that the set of deleted weight-1 voters corresponds to a cover. □

4.3 Approximation and Greedy Algorithms

When problems are computationally difficult, such as being NP-complete, it is natural to wonder whether good polynomial-time approximation algorithms exist. So, motivated by the NP-completeness results discussed earlier in this paper for most cases of WCCAV/WCCDV for t-approval and t-veto, this section studies greedy and other approximation algorithms for those problems. (Recall that WCCAV is NP-complete for t-approval, $t \geq 3$, and for t-veto, $t \geq 2$, and WCCDV is NP-complete for t-approval, $t \geq 2$, and for t-veto, $t \geq 3$.) First, we will establish a connection to the weighted multicover problem, and we will use it to obtain approximation results. Then we will obtain an approximation algorithm that will work by direct action on our problem. Table 3 in Section 6 summarizes our results on approximation algorithms for t-approval/t-veto WCCAV/WCCDV.

Before we undertake this, let us address in more detail the issue, valuably raised by a referee, of why one might want to build approximation algorithms for control problems, and who might use such algorithms, and whether it is unwise to obtain such algorithms if the people using them might not be "the good guys." As mentioned above, seeking good polynomial-time approximation algorithms is one standard approach when exact solutions are known to be intractable, e.g., NP-complete. Such algorithms will allow a campaign strategist to, faced with the intractability of computing the optimal number of votes to add or delete to achieve victory for his or her candidate, at least be able to quickly find an action that is guaranteed to be within a particular multiplicative factor of the optimal action. One might expect that our desire to get such approximations would hit a wall regarding the potential impossibility of exerting control in certain instances, but as we discussed in the Electoral Control subpart of Section 3, that worry does not hold for the particular problems for which we will obtain approximation algorithms. Finally, as to the worry that people ("chairs") who employ approximation algorithms may not be "the good guys," we have the following somewhat multilayered reply. First, "good" and "evil" are highly contextual. Whether a strategist's attempts to help his or her candidate win are good or evil is very much in the eye of the beholder. Some may decry such attempts as part of the brutal nature of politics. Others may view such attempts, as long as no illegal actions are taken, as a valid and indeed valuable part of the spirited, vibrant playing field of democracy. Second, in some settings, control may be simply modeling an optimization problem, and so well-approximating control isn't even about candidates, but is simply about efficiency. Third, even if one views approximating control as helping evil-doers, using that as a reason not to learn which control problems can be approximated and how well they can be approximated makes no more sense than sticking one's head in the sand and hoping that cryptosystems can't be broken. Since evil-doers may well try to build approximation algorithms, or break cryptosystems, the natural way of thwarting them is for the field to richly explore what approximations and vulnerabilities exist, so that those who choose what election system to use for a given problem can choose one that is not weak with respect to having good approximations under whatever attacks they most fear.

4.3.1 A Weighted Multicover Approach

Let us first consider the extent to which known algorithms for the Set-Cover family of problems apply to our setting. Specifically, we will use the following multicover problem.

Definition 4.14. *An instance of Weighted Multicover (WMC) consists of a set* $B = \{b_1, \ldots, b_m\}$*, a sequence* $r = (r_1, \ldots, r_m)$ *of nonnegative integers (covering requirements), a collection* $\mathcal{S} = (S_1, \ldots, S_n)$ *of subsets of* B*, and a sequence* $\omega = (\omega_1, \ldots, \omega_n)$ *of positive integers (weights of the sets in* $\mathcal{S}$*). The goal is to find a minimum-cardinality set* $I \subseteq \{1, \ldots, n\}$ *such that for each* $b_j \in B$ *it holds that* $r_j \leq \sum_{i \in I \wedge b_j \in S_i} \omega_i$*, or to declare that no such set exists.*

That is, given a WMC instance we seek a smallest collection of subsets from $\mathcal{S}$ that satisfies the covering requirements of the elements of B (keeping in mind that a set of weight ω covers each of its elements ω times). WMC is an extension of Set-Cover with unit costs. We will not define here the problem known as Covering Integer Programming (see Kolliopoulos & Young, 2005), which for short is written as CIP. However, that problem will be quite important to us here. The reason is that we observe that WMC is a special case of CIP (with multiplicity constraints but) without packing constraints; footnote 6 below is in effect describing how to embed our problem in that problem. An approximation algorithm of Kolliopoulos and Young for CIP (with multiplicity constraints but) without packing constraints, applied to the special case of WMC, gives the following result.[6]

Theorem 4.15 (Kolliopoulos & Young, 2005)**.** *There is a polynomial-time algorithm that when given an instance of WMC in which each set contains at most* t *elements gives an* $\mathcal{O}(\log t)$*-approximation.*

For t-approval both WCCAV and WCCDV naturally translate to equivalent WMC instances. We consider WCCAV first. Let (C, V, W, p, k) be an instance of t-approval-

6. The paper of Kolliopoulos and Young (2005) does not directly speak of the WMC problem, but seeing that their results indeed apply to WMC is an easy, if tedious, exercise. For those readers who would like to verify that Theorem 4.15 holds, in this footnote we describe exactly where in the paper of Kolliopoulos and Young one finds the relevant result and which parameters one should use. We warn the reader that this footnote makes direct references to parts of that paper and so will only make sense if that paper is simultaneously in hand. This footnote is merely a guide to understanding the particular way we draw on that paper's important work; providing a full-fledged survey of, or even a real discussion of, the CIP problem is beyond the needs and scope of this paper.

   Theorem 4.15 follows from the sentence—on page 496 of the work of Kolliopoulos and Young (2005)—starting "Our second algorithm finds a solution" (which itself follows from their Theorem 8), keeping in mind that we have none of their so-called packing constraints, and so we may take it that what they call ϵ is one and the matrix and vector they call B and b won't be a factor here. Their vector a corresponds to our r_j's; the element in the jth row and ith column of their matrix A will for us be set to ω_i if S_i contains b_j and 0 otherwise; we set their cost vector c to be a vector of all 1's; we set their multiplicity vector d to be a vector of all 1's; their vector x corresponds to the characteristic function of our I; and their α will be Theorem 4.15's bound t on the number of elements of B contained in any S_i.

WCCAV, where $W = (w_1, \ldots, w_n)$ is the collection of voters that we may add. We assume without loss of generality that each voter in W ranks p among its top t candidates (i.e., approves of p).

We form an instance $(B, r, \mathcal{S}, \omega)$ of WMC as follows. We set $B = C - \{p\}$. For each $c \in B$, we set its covering requirement to be $r_c = \mathit{score}_{(C,V)}(c) \ominus \mathit{score}_{(C,V)}(p)$, where $i \ominus j =_{def} \max(0, i-j)$. For each vote $w \in W$, let S_w be the set of candidates that w does not approve of. By our assumption regarding each voter ranking p among its top t candidates, no S_w contains p. We set $\mathcal{S} = (S_{w_1}, \ldots, S_{w_n})$ and we set $\omega = (\omega(w_1), \ldots, \omega(w_n))$. It is easy to see that a set $I \subseteq \{1, \ldots, n\}$ is a solution to this instance of WMC (that is, I satisfies all covering requirements) if and only if adding the voters $\{w_i \mid i \in I\}$ to the election (C, V) ensures that p is a winner. The reason for this is the following: If we add voter w_i to the election then for each candidate $c \in S_{w_i}$, the difference between the score of c and the score of p decreases by $\omega(w_i)$, and for each candidate $c \notin S_{w_i}$ this difference does not change. The covering requirements are set to guarantee that p's score will match or exceed the scores of all candidates in the election.

We stress that in the above construction we did not assume t to be a constant. Indeed, the construction applies to t-veto just as well as to t-approval. So using Theorem 4.15 we obtain the following result.

Theorem 4.16. *There is a polynomial-time $\mathcal{O}(\log m)$-approximation algorithm for t-approval-WCCAV. There is a polynomial-time algorithm that when given an instance of t-veto-WCCAV ($t \in \mathbb{N}$) gives an $\mathcal{O}(\log t)$-approximation.*

Proof. It suffices to use the reduction of t-approval/t-veto to WMC and apply the algorithm from Theorem 4.15. For the case of t-approval, the reduction guarantees that each set in the WMC instance contains at most m elements. For the case of t-veto, each of these sets contains at most t elements. ☐

We can obtain analogous results for the case of t-approval/t-veto and WCCDV. One can either provide a direct reduction from these problems to WMC or notice that the reductions given in the proof of Theorem 4.12 maintain approximation properties.

Theorem 4.17. *There is a polynomial-time algorithm that when given an instance of t-approval-WCCDV ($t \in \mathbb{N}$) gives an $\mathcal{O}(\log t)$-approximation. There is a polynomial-time $\mathcal{O}(\log m)$-approximation algorithm for t-veto-WCCDV.*

4.3.2 A Direct Approach

Using algorithms for WMC, we were able to obtain relatively strong algorithms for WCCAV/WCCDV under t-approval and t-veto. However, with this approach we did not find approximation algorithms for t-approval-WCCAV and t-veto-WCCDV whose approximation ratios depend only on t (and not, for example, on $\|C\|$, i.e., m, or on $\|V\|$). In the following we will seek direct algorithms for these problems.

We now show that a very simple greedy approach yields a polynomial-time t-approximation algorithm for t-approval-WCCAV and t-veto-WCCDV. (Recall that this means that in cases when making p win is possible, the number of voters our algorithm adds/deletes to reach victory is never more than t times that of the optimal set of additions/deletions.)

Let GBW (greedy by weight) define the following very simple algorithm for WCCAV. (The votes are the weighted t-approval vectors induced by the preferences of the voters.) (Pre)discard all unregistered votes that do not approve of the preferred candidate p. Order the (remaining) unregistered votes from heaviest to lightest, breaking ties in voter weights in some simple, transparent way (for concreteness, let us say by lexicographic order on the votes' representations). GBW goes through the unregistered votes in that order, and as it reaches each vote it adds the vote exactly if the vote disapproves of at least one candidate whose score (i.e., total weight of approvals) is currently strictly greater than that of p. It stops successfully when p has become a winner and unsuccessfully if before that happens the algorithm runs out of votes to consider. The following result says that GBW is a t-approximation algorithm for t-approval-WCCAV, and also for t-veto-WCCDV, using the obvious analogue of GBW for t-veto-WCCDV, which we will also call GBW.[7]

Theorem 4.18. *Let $t \geq 3$. The polynomial-time greedy algorithm GBW is a t-approximation algorithm for t-approval-WCCAV and t-veto-WCCDV; and there are instances in which GBW's approximation factor on each of these problems is no better than t.*

We prove Theorem 4.18's upper and lower bound parts separately, through the following two lemmas from which the theorem immediately follows.

Lemma 4.19. *Let $t \geq 3$. There are instances on which the polynomial-time greedy algorithm GBW has an approximation factor on t-approval-WCCAV no better than t. There are instances on which the polynomial-time greedy algorithm GBW has an approximation factor on t-veto-WCCDV no better than t.*

Lemma 4.20. *Let $t \geq 3$. The polynomial-time greedy algorithm GBW is a t-approximation algorithm for t-approval-WCCAV and t-veto-WCCDV.*

The proof of our lower-bound claim, Lemma 4.19, consists of a somewhat detailed pair of constructions, and is of less interest than the upper-bound part of Theorem 4.18, namely Lemma 4.20. We thus defer to the appendix the proof of Lemma 4.19.

Proof of Lemma 4.20. Let us now prove the two claims that GBW is a t-approximation algorithm. We will prove the result for $t = 3$ and WCCAV, but it will be immediately clear that our proof straightforwardly generalizes to all greater t; and the WCCDV case follows using Theorem 4.12.

Clearly GBW is a polynomial-time algorithm. Consider a given input instance of t-approval-WCCAV, with preferred candidate p. Without loss of generality, assume all unregistered voters approve of p. We will say a candidate "has a gap" (under the current set of registered voters and whatever unregistered voters have already been added) if that candidate has strictly more weight of approvals than p does. For each candidate d who has

7. For completeness and clarity, we describe what we mean by GBW for t-veto-WCCDV. Order all votes that do not approve of p from heaviest to lightest, breaking ties in voter weights in some simple, transparent way (for concreteness, let us say by lexicographic order on the votes' representations). GBW goes through these votes in that order, and as it reaches each vote it removes the vote exactly if the vote approves of at least one candidate whose score (i.e., total weight of approvals) is currently strictly greater than that of p. It stops successfully when p has become a winner and unsuccessfully if before that happens the algorithm runs out of such votes to consider.

a gap, $d \neq p$, define i_d to be the minimum number of unregistered voters one has to add to remove d's gap; that is, if one went from heaviest to lightest among the unregistered voters, adding in turn each that disapproved of d, i_d is the number of voters one would add before d no longer had a gap. If for any candidate d it holds that no integer realizes i_d, then control is impossible using the unregistered voter set. Clearly, any successful addition of voters must add at least $\max_d i_d$ voters (the max throughout this proof is over all candidates initially having a gap).

Let us henceforth assume that control is possible in the input case. We will show that after having added at most $3 \cdot \max_d i_d$ voters GBW will have made p a winner, and so GBW is a 3-approximation algorithm.

Before giving the detailed proof, let us very informally give a sense of the proof's idea. Let z be some candidate who allegedly has a gap after GBW has just added $3 \cdot \max_d i_d$ voters, and freeze the action of GBW at that point. Our proof argues that if relatively many of the $3 \cdot \max_d i_d$ voters added by GBW (i.e., at least $\max_d i_d$ of them) do not approve of z, then z clearly will not have a gap at the point in time when GBW was frozen, and so the assumed gap can't exist in this case. Our proof further argues that if relatively few of the $3 \cdot \max_d i_d$ voters added by GBW (i.e., at most $(\max_d i_d) - 1$ of them) do not approve of z (equivalently, at least $1 + 2 \cdot \max_d i_d$ of them *do* approve of z), then we also arrive at a contradiction. The latter argument is a more subtle one, involving asking which candidate's (call it y) gap caused the very last added vote to be added, and if needed drilling down an extra level with a related few/many argument now focused on y, to show that GBW must at some point have acted in a way that violates its definition, thus also yielding a contradiction. Since the "few" and "many" cases above cover all possible cases, our proof will have achieved its goal. We provide now the formal analysis that carries out this argument line.

So, suppose that after $3 \cdot \max_d i_d$ additions some candidate, z, still has a gap. As discussed above we will perform a case analysis to in each case arrive at a contradiction.

Case 1 [In at least $\max_d i_d$ *of the first* $3 \cdot \max_d i_d$ *votes added by GBW, z is not approved].* Since for the last one of these to be added z must still have had a gap before the addition, each earlier vote considered that disapproved z had a gap for z when it was considered and so would have been added when reached. So, keeping in mind that $i_z \leq \max_d i_d$, we in fact must have added the i_z heaviest voters disapproving of z, and so contrary to the assumption, z no longer has a gap after these additions.

Case 2 [Case 1 does not hold]. So z is approved in at least $1 + 2 \cdot \max_d i_d$ of the added votes. What made the final one of the added votes, call it v', eligible for addition? It must be that some candidate, say y, still had a gap just before v' was added.

Case 2a [y is disapproved in at least $\max_d i_d$ *of the* $2 \cdot \max_d i_d$ *votes added before* v' *that approved z].* Then, since until y's gap was removed no unregistered voters disapproving of y would be excluded by GBW, y's i_y heaviest voters will have been added. So contrary to Case 2's assumption, y does not have a gap when we get to adding vote v'.

Case 2b [Case 2 holds but Case 2a does not]. Then y is approved in at least $1 + \max_d i_d$ of the $2 \cdot \max_d i_d$ votes before v' that GBW added that approve z. So we have $1 + \max_d i_d$ votes added approving of exactly z and y. But then who made the last of *those* $1 + \max_d i_d$ votes, call it v'', eligible to be added? It must hold that some candidate w had a gap up through v''. But at the moment before adding v'' we would have added $\max_d i_d \geq i_w$ votes approving exactly z and y and so disapproving w, and since w allegedly still had a gap, we while doing so under GBW would have in fact added the i_w heaviest voters disapproving of w, and so w's gap would have been removed before v'', so contrary to our assumption w was not the gap that made v'' eligible. ☐

One might naturally wonder how GBW performs on t-veto-WCCAV and t-approval-WCCDV. By an argument far easier than that used in the above proof of Lemma 4.20, in both of these cases GBW provides a t-approximation algorithm.

Theorem 4.21. *GBW is a t-approximation algorithm for t-veto-WCCAV. GBW is a t-approximation algorithm for t-approval-WCCDV.*

Proof. Consider t-veto-WCCAV. Let p be the preferred candidate. For each candidate d with an initial positive "gap" relative to the preferred candidate p (i.e., a surplus over p in total weight of approvals), let i_d be as defined in the proof of Lemma 4.20. (Recall that i_d is the number of votes we would need to add to remove the surplus of d over p if we took the unregistered votes, discarded all that didn't simultaneously approve p and disapprove d, and then added those one at a time from heaviest to lightest until the gap was removed.) Clearly, $\sum i_d$, where the sum is taken over those candidates with an initial surplus relative to p, is an upper bound on the number of votes added by GBW. This is true since GBW works by adding extra votes from heaviest to lightest, restricted to those vetoing a candidate who at that point has a positive gap relative to p; so under GBW each gap will be closed by the largest weight votes that address it. On the other hand, in any overall optimal solution i_d is a lower bound on the smallest number of votes from that solution's added-vote set that would suffice to remove d's positive gap (since it takes i_d even if we use the heaviest votes addressing the gap). In the overall optimal solution each added vote narrows at most t gaps. So GBW's solution uses at worst t times as many added votes as does the optimal solution.

The claim for t-approval-WCCDV follows by Theorem 4.12. ☐

This result replaces a flawed claim in the conference version of this paper (Faliszewski, Hemaspaandra, & Hemaspaandra, 2013) that GBW and some of its cousins do not provide $\mathcal{O}(1)$ approximations for these problems.[8] Of course, having a t-approximation for these two problems (namely, t-veto-WCCAV and t-approval-WCCDV) is not wildly exciting, since for these problems the multicover-based approach from earlier in this section showed that for some function $f(t)$, with $f(t) = \mathcal{O}(\log t)$, we even have $f(t)$-approximation algorithms for these problems. However, if the constant of the "big oh" of that other algorithm is large, it is possible that for sufficiently small values of t the above approach may give a better approximation. Also, we feel that it is interesting to learn about the behavior of explicit heuristics, especially attractive approaches such as greedy algorithms.

It is natural to ask whether similar greedy algorithms work well for other scoring rules, e.g., for Borda's rule. Unfortunately, for families of scoring rules other than t-approval and t-veto the analysis, if at all possible, would likely have to be significantly different than ours. The main reason for this is that—as discussed in the Electoral Control subpart of

8. Note that here we treat t as a constant and, so, a t-approximation algorithm provides (indeed, is) an $\mathcal{O}(1)$ approximate one. The reason that is true is that, technically speaking, the WCCAV and WCCDV problems are defined separately for each voting rule. For example, 2-approval-WCCAV is a different problem than, say, 200-approval-WCCAV.

Section 3—for t-approval and t-veto it is always easy to verify whether there exists *some* solution (although, perhaps, one that is very far from being optimal). For other scoring rules, e.g., for Borda, it is not at all clear whether this is possible (and we conjecture that, indeed, it is NP-complete to do so). However, it might be an interesting research direction to evaluate the effectiveness of such greedy algorithms empirically (we point the reader to the work of Rothe & Schend, 2013, for a recent survey covering experimental studies of the complexity of control in elections).

5. Related Work

The study of the complexity of (unweighted) electoral control was initiated by Bartholdi, Tovey, and Trick (1992), who considered constructive control by adding/deleting/partitioning candidates/voters under the plurality rule and under the Condorcet rule (that is, the rule that chooses Condorcet winner whenever there is one, and has no winners otherwise). The various types of control model at least some of the flavor of actions that occur in the real world, such as voter suppression and targeted get-out-the-vote drives (see the survey of Faliszewski et al., 2010, for more examples and discussions). A major motivation for the study of control was to obtain "complexity barrier" results, that is, results that show that detecting opportunities for various control attacks is computationally difficult. In particular, Bartholdi, Tovey, and Trick focused on NP-hardness as the measure of computational difficulty.

This research direction was continued by Hemaspaandra, Hemaspaandra, and Rothe (2007), who were the first to study destructive control attacks on elections. Since then, many authors have studied electoral control in many varied settings and under many different rules; we refer the reader to the survey of Faliszewski et al. (2010). Some recent research, not covered in that survey, includes complexity-of-control results for the t-approval family of rules (Lin, 2012), for Bucklin's rule (and for fallback, its extension for truncated votes; Erdélyi, Fellows, Rothe, & Schend, 2015a), for maximin (Faliszewski et al., 2011), for range voting (Menton, 2013), and for Schultze's rule and the ranked pairs rule (Parkes & Xia, 2012; Menton & Singh, 2013; Hemaspaandra, Lavaee, & Menton, 2013). In the present paper, we compare control and manipulation. The recent paper of Fitzsimmons, Hemaspaandra, and Hemaspaandra (2013) studies settings in which both control and manipulation are occurring. Researchers have, in the quite different setting of electing members to fill a fixed-size, multimember panel, defined variants of control that have coexisting constructive and destructive aspects (Meir, Procaccia, Rosenschein, & Zohar, 2008). There is also work analyzing counting variants of control (Wojtas & Faliszewski, 2012), where the goal is not only to decide if a given control attack is possible, but also to count the number of ways in which this attack can be carried out.

The complexity-barrier research line turned out to be very successful. For most voting rules that were considered, a significant number of control attacks are NP-hard. Indeed, it is even possible to construct an artificial election system resistant to all types of control attacks (Hemaspaandra, Hemaspaandra, & Rothe, 2009). However, there are also a number of results that suggest that in practice the complexity barrier might not be as strong as one might at first think. For example, Faliszewski, Hemaspaandra, Hemaspaandra, and Rothe (2011) and Brandt, Brill, Hemaspaandra, and Hemaspaandra (2010) have shown that if the votes are restricted to being single-peaked, then many control problems that are known to be NP-complete become polynomial-time solvable. Indeed, this often holds even if elections are just nearly single-peaked (Faliszewski et al., 2014), as many real-world elections seem to be (see, e.g., the discussion in Gehrlein & Lepelley, 2012, ch. 2). Similarly, some initial experimental results of Erdélyi, Fellows, Rothe, and Schend (2015b) suggest that, at least under certain distributions and settings, some NP-hard control problems can be solved in practice on many instances. As part of a different line of research, Xia (2012) has studied the asymptotic behavior of the number of voters that have to be added to/deleted from a randomly constructed election in a successful control action.

There are a number of other problems involving changing the structure of elections. These problems include candidate cloning, where it is possible to replace a given candidate c with a number of its clones (Elkind et al., 2011; Elkind, Faliszewski, & Slinko, 2012), or the possible winner problem when new alternatives join, where some additional, not yet ranked candidates can be introduced (Chevaleyre, Lang, Maudet, Monnot, & Xia, 2012; Xia, Lang, & Monnot, 2011). This last problem is also related to the possible winner problem with truncated ballots (Baumeister, Faliszewski, Lang, & Rothe, 2012a).

The only papers that directly raise the issue of weighted control are, to the best of our knowledge, the theses of Russell (2007) and Lin (2012). However, we also mention the papers of Baumeister, Roos, Rothe, Schend, and Xia (2012b), and of Perek, Faliszewski, Pini, and Rossi (2013), where the authors, in effect, consider problems of affecting the result of an election through picking the weights of the voters. (The paper of Perek et al. motivates its study differently, but in effect studies a constrained variant of choosing voter weights.) Their problems are similar to, though different from, simultaneous (multimode) addition and deletion of voters (Faliszewski et al., 2011).

This paper has given $f(\cdot)$-approximation results for weighted election control problems. Elkind and Faliszewski (2010) have given a 2-approximation algorithm for a weighted, bribery-related case.

6. Conclusions

We have studied voter control under a number of voting rules, including scoring protocols, families of scoring protocols, and the (weak)Condorcet-consistent rules. We have shown that the complexity of voter control can be quite different from the complexity of weighted coalitional manipulation: there are natural voting rules for which weighted coalitional manipulation is easy but weighted voter control is hard, and there are natural rules where the opposite is the case. Furthermore, we have shown that for weighted voter control under t-approval and t-veto, there are good, natural approximation algorithms. Our results for voter control in weighted elections are summarized in Tables 1, 2, and 3.

Acknowledgements

We are very grateful to the anonymous AAMAS 2013 and *JAIR* referees for extremely helpful comments and suggestions, some of which we have incorporated as examples. We thank the editor, Jérôme Lang, for his wise guidance. This work was sup-

	WCCAV	WCCDV	WCM
Plurality	P (Thm. 4.1)	P (Thm. 4.1)	P♣
t-approval, $2 \leq t < m$	P (Thm. 4.1)	P (Thm. 4.1)	NP-comp.♢
Borda	NP-comp. (Thm. 4.2)	NP-comp. (Thm. 4.2)	NP-comp.♢
$\alpha = (\alpha_1, \ldots, \alpha_m)$, $\|\{\alpha_1, \ldots, \alpha_m\}\| \geq 3$	NP-comp. (Thm. 4.3)	NP-comp. (Thm. 4.3)	NP-comp.♢
Llull (3 candidates)	NP-comp. (Cor. 4.9)	NP-comp. (Cor. 4.9)	P♡
(weak)Condorcet-consistent rules	NP-hard (Thm. 4.7)	NP-hard (Thm. 4.7)	various complexities

Table 1: Our results for the complexity of control by adding/deleting voters in weighted elections for any fixed number of candidates, $m \geq 3$, compared to the complexity of weighted coalitional manipulation. The result marked with ♣ is due to Conitzer et al. (2007), the results marked with ♢ are due to Hemaspaandra and Hemaspaandra (2007), and the result marked with ♡ is due to Faliszewski et al. (2008).

	WCCAV	WCCDV
t-approval		
$t = 2$	P (Thm. 4.10)	NP-complete (Thm. 4.13)
$t = 3$	NP-complete (Thm. 4.13)	NP-complete♠
$t \geq 4$	NP-complete♠	NP-complete♠
t-veto		
$t = 2$	NP-complete (Thm. 4.13)	P (Thm. 4.11)
$t = 3$	NP-complete♠	NP-complete (Thm. 4.13)
$t \geq 4$	NP-complete♠	NP-complete♠

Table 2: The complexity of control by adding and deleting voters for t-approval and t-veto with an unbounded number of candidates. The results marked with ♠ are due to Lin (2012).

ported in part by grants AGH-11.11.230.124, NCN-DEC-2011/03/B/ST6/01393, NCN-UMO-2012/06/M/ST1/00358, and NSF-CCF-{0915792,1101452,1101479}, and two Bessel Awards from the Alexander von Humboldt Foundation.

	WCCAV	WCCDV
t-approval	$\mathcal{O}(\log m)$ (Thm. 4.16) t (Thm. 4.18)	$\mathcal{O}(\log t)$ (Thm. 4.17) t (Thm. 4.21)
t-veto	$\mathcal{O}(\log t)$ (Thm. 4.16) t (Thm. 4.21)	$\mathcal{O}(\log m)$ (Thm. 4.17) t (Thm. 4.18)

Table 3: Approximation ratios of our algorithms for WCCAV and WCCDV under t-approval and t-veto.

Appendix A. Additional Details Related to Section 4.3

We present here the deferred proof of Lemma 4.19 and some other details related to Section 4.3.

Proof of Lemma 4.19. Our goal is to show that GBW sometimes really does use fully t times the optimal number of added/deleted votes, for the cases in question. Examples are (somewhat detailed but) not hard to construct, and the lower bound even holds for $t = 2$, though in Section 4.2 we obtained an exact solution by a different approach. However, one does have to be careful to set the "gap" pattern created by the unregistered voters to be a realizable one. For our t-approval-WCCAV construction, this will be easy to do directly. For our t-veto-WCCDV construction, we will establish realizability through a small tool—which we hope may prove useful elsewhere—that lets one set up certain patterns of gaps. We state the tool below as Tool A.1.

Fix any $t \in \{2, 3, 4, \ldots\}$. We will now construct an instance of t-approval-WCCAV on which GBW uses t times as many additions as the optimal strategy. Our construction will have $2t$ candidates: the preferred candidate p, candidates $a_1, \ldots, a_t$, and candidates $d_1, \ldots, d_{t-1}$. Now, suppose that under the votes of the registered voters, the "gaps" are as follows. For each candidate a_i, the total weight of approvals of a_i exceeds the total weight of approvals of p by exactly $2t$. And for each candidate d_i, the total weight of approvals of d_i equals the total weight of approvals of p. This can easily be realized, namely by our registered voter set being one weight-$2t$ voter who approves of each a_i.

Our set of unregistered voters will be as follows. There will be one unregistered voter, call it "nice," of weight $2t$, who approves of p and each of the $t-1$ candidates d_i, and disapproves of each of the t candidates a_i. For each j, $1 \leq j \leq t$, we will have a single unregistered voter, call it α_j, of weight $3t$, who approves of p and of each a_i other than a_j, and disapproves of a_j and all the d_i's.

Note that GBW will add all t voters α_i. But ideal would be to add the single voter called "nice," since doing so suffices to make p a winner. So for each $t \geq 2$ we have constructed a setting where GBW for t-approval-WCCAV takes t times more than the optimal number of added votes.

It also holds that for each $t \geq 2$, we can similarly construct a setting where GBW for t-veto-WCCDV takes t times more than the optimal number of deleted votes, and can prove that setting to be realizable. In fact, we can do so by following something of the flavor of the above scheme, except with a slightly different vote set that adjusts it to handle the case of deleting voters, and with more care regarding realizability. Here is the construction. Fix any $t \in \{2, 3, 4, \dots\}$. Our candidate set will again be the preferred candidate p, candidates $a_1, \dots, a_t$, and candidates $d_1, \dots, d_{t-1}$. Let us specify the voter set. We will put into our voter set a collection of weight-1 votes such that the gaps in total approval weight *relative to* d_1 created by those votes are as follows. Each of d_2 through d_{t-1} have the same total approval weight as d_1. The total approval weight of p exceeds that of d_1 by $3t^2 + 3t$. And the total approval weight of each a_i exceeds that of d_1 by $3t^2$.

As Tool A.1 below, we will observe that for $2t$-candidate t-approval voting, any gap pattern where the gaps are all multiples of t can be realized. Since in the current proof we are using $2t$-candidate t-veto, and that is the same as $2t$-candidate t-approval, Tool A.1 applies here. In particular, Tool A.1 easily builds a set of weight-1 votes realizing precisely our desired set of gaps. (The exact number of weight-1 votes used in this construction is not important. However, from the gaps mentioned above and the vote-set size mentioned in the tool, the precise number is easily seen to be $(3t + 3 + t(3t))(2t - 1)$.)

We are not yet done building our voter set. We will also have in our voter set one voter, call it "nice," of weight $2t$, who approves of exactly all t of the a_i's. And for each j, $1 \leq j \leq t$, we will have one voter of weight $3t$ who approves of exactly a_j and all $t - 1$ of the d_i's.

Under the entire set of votes created above—the votes from the tool combined with "nice" and the other t votes just mentioned—it is easy to see that d_1 is a candidate having the least total approval weight, and it is tied in total approval weight with each other d_i. The total approval weight of p exceeds that of d_1 by $3t$. And each a_i exceeds d_1 in total approval weight by $5t$.

However, in light of the pattern of votes and weights we have here, it is clear that GBW (in its version for t-veto) will delete the t weight-$3t$ voters. (Note that the votes added by Tool A.1 are all weight-1 votes, and so are highly unattractive to GBW.) But ideal would be to delete the single voter called "nice," since doing so suffices to make p a winner. So for each $t \geq 2$ we have constructed a realizable setting where GBW for t-veto-WCCDV takes t times more than the optimal number of deleted votes. $\square$

Within the above proof, we referred to and used a small tool that can build certain patterns of vote weight gaps in certain approval elections. It would be an overreach to claim that this is a McGarvey-like tool, since this is a different setting than, and is a far less flexible result than, the famous theorem of McGarvey (1953). However, it in a small way is a tool that perhaps might be useful elsewhere, and so we state and prove this modest tool.

Tool A.1. *Let $t \geq 2$. Let $n_1, \dots, n_{2t-1}$ be any list of nonnegative integers each divisible by t. Then there exists a collection of t-approval votes, over $2t$ candidates, such that under those votes, relative to the candidate getting the fewest approvals, the list of gaps in number of approvals between that candidate and the other $2t-1$ candidates is precisely $(n_1, \dots, n_{2t-1})$. Furthermore, this can be done with $(2t - 1)(\sum n_i)/t$ unweighted (i.e., weight 1) votes. It alternatively can be done with $(2t - 1)^2$ weighted votes (or even $(2t - 1)\|\{i \mid n_i \neq 0\}\|$ weighted votes).*

Proof. Consider an election with $2t$ candidates, where the votes cast are t-approval votes. Consider the collection of $2t - 1$ votes, each of weight one, in which the votes all approve of a particular candidate (for this example, let that one be the first candidate), and the remaining $t-1$ approvals cyclically rotate around the other candidates. So the t-approval votes, viewed as bit vectors, are these: $1\,1^{t-1}\,0^t$, $1\,0\,1^{t-1}\,0^{t-1}$, $\ldots$, $1\,0^t\,1^{t-1}$, $1\,1\,0^t\,1^{t-2}$, $\ldots$, $1\,1^{t-1}\,0^t\,1$. Note that the first candidate is approved in all $2t-1$ of those votes, and each other candidate is approved in exactly $t-1$ of those votes. So this collection of votes sets a gap of t in favor of the first candidate, between the total approval weight of the first candidate and that of each other candidate And the difference in total approval weight between each other pair of candidates is zero.

Given a gap pattern as stated in the tool, where each gap above the least-approved candidate (call that candidate c) is a multiple of t, we can simply use the approach of the above paragraph repeatedly, to boost each other candidate, d, one at a time to whatever multiple of t it is supposed to exceed c by in total approval weight. (In this, d will play the role "the first candidate" did in the previous paragraph.) If d's surplus relative to c is kt and we wish to use only weight-1 votes, we can do this for d with $k(2t-1)$ weight-1 votes. Otherwise, we can do this for d with $2t-1$ weight-k votes. So the total number of votes used is as given in the statement of this tool. $\square$

This appendix is not seeking to provide a comprehensive study of which gap collections are realizable under t-approval voting, nor is it seeking to find the smallest number of voters needed to realize realizable gap collection. That is an interesting direction for study, but is not our goal here. However, we mention that there clearly exist some gap collections that cannot be realized. For example, the "then there exists" claim of Tool A.1 is not even always true if one removes the assumption of divisibility by t. An example showing this is the following. Consider a 4-candidate setting where votes will be 2-approval votes, and we desire a gap list relative to the least-approved candidate of $(1, 1, 1)$, i.e., each of the other candidates has one more approval than does the least-approved candidate. Clearly, the total number of approvals of any set of votes achieving this is $4B + 3$, where B is whatever number of approvals the least-approved candidate happens to get under the vote set one is trying, and so the total number of approvals is odd. However, any vote set of 2-approval votes has an even total number of approvals. So this gap collection cannot be realized.

References

Aziz, H., Bachrach, Y., Elkind, E., & Paterson, M. (2011). False-name manipulations in weighted voting games. *Journal of Artificial Intelligence Research*, *40*, 57–93.

Bartholdi, III, J., & Orlin, J. (1991). Single transferable vote resists strategic voting. *Social Choice and Welfare*, *8*(4), 341–354.

Bartholdi, III, J., Tovey, C., & Trick, M. (1989). The computational difficulty of manipulating an election. *Social Choice and Welfare*, *6*(3), 227–241.

Bartholdi, III, J., Tovey, C., & Trick, M. (1992). How hard is it to control an election?. *Mathematical and Computer Modeling*, *16*(8/9), 27–40.

Baumeister, D., Faliszewski, P., Lang, J., & Rothe, J. (2012a). Campaigns for lazy voters: Truncated ballots. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pp. 577–584.

Baumeister, D., Roos, M., Rothe, J., Schend, L., & Xia, L. (2012b). The possible winner problem with uncertain weights. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pp. 133–138.

Brandt, F., Brill, M., Hemaspaandra, E., & Hemaspaandra, L. (2010). Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pp. 715–722.

Brandt, F., Conitzer, V., & Endriss, U. (2013). Computational social choice. In Weiß, G. (Ed.), *Multiagent Systems* (2nd edition). MIT Press.

Brelsford, E., Faliszewski, P., Hemaspaandra, E., Schnoor, H., & Schnoor, I. (2008). Approximability of manipulating elections. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp. 44–49. AAAI Press.

Chen, J., Faliszewski, P., Niedermeier, R., & Talmon, N. (2014). Combinatorial voter control in elections. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science, Part II*, pp. 153–164. Springer-Verlag *Lecture Notes in Computer Science #8635*.

Chevaleyre, Y., Lang, J., Maudet, N., Monnot, J., & Xia, L. (2012). New candidates welcome! Possible winners with respect to the addition of new candidates. *Mathematical Social Sciences*, *64*(1), 74–88.

Congleton, R. (2011). The Swedish transition to democracy (Chapter 14). In *Perfecting Parliament*. Cambridge University Press.

Conitzer, V., Sandholm, T., & Lang, J. (2007). When are elections with few candidates hard to manipulate?. *Journal of the ACM*, *54*(3), Article 14.

Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). Rank aggregation methods for the web. In *Proceedings of the 10th International World Wide Web Conference*, pp. 613–622. ACM Press.

Elkind, E., & Faliszewski, P. (2010). Approximation algorithms for campaign management. In *Proceedings of the 6th International Workshop On Internet And Network Economics*, pp. 473–482.

Elkind, E., Faliszewski, P., & Slinko, A. (2011). Cloning in elections: Finding the possible winners. *Journal of Artificial Intelligence Research*, *42*, 529–573.

Elkind, E., Faliszewski, P., & Slinko, A. (2012). Clone structures in voters' preferences. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pp. 496–513.

Ephrati, E., & Rosenschein, J. (1997). A heuristic technique for multi-agent planning. *Annals of Mathematics and Artificial Intelligence*, *20*(1–4), 13–67.

Erdélyi, G., Fellows, M., Rothe, J., & Schend, L. (2015a). Control complexity in Bucklin and fallback voting: A theoretical analysis. *Journal of Computer and System Sciences*, *81*(4), 632–660.

Erdélyi, G., Fellows, M., Rothe, J., & Schend, L. (2015b). Control complexity in Bucklin and fallback voting: An experimental analysis. *Journal of Computer and System Sciences*, *81*(4), 661–670.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2009). How hard is bribery in elections?. *Journal of Artificial Intelligence Research*, *35*, 485–532.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2010). Using complexity to protect elections. *Communications of the ACM*, *53*(11), 74–82.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2011). Multimode attacks on elections. *Journal of Artificial Intelligence Research*, *40*, 305–351.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2013). Weighted electoral control. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pp. 367–374.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2014). The complexity of manipulative attacks in nearly single-peaked electorates. *Artificial Intelligence*, *207*, 69–99.

Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2009). A richer understanding of the complexity of election systems. In Ravi, S., & Shukla, S. (Eds.), *Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz*, pp. 375–406. Springer.

Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2011). The shield that never was: Societies with single-peaked preferences are more open to manipulation and control. *Information and Computation*, *209*(2), 89–107.

Faliszewski, P., Hemaspaandra, E., & Schnoor, H. (2008). Copeland voting: Ties matter. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, pp. 983–990. International Foundation for Autonomous Agents and Multiagent Systems.

Fitzsimmons, Z., Hemaspaandra, E., & Hemaspaandra, L. (2013). Control in the presence of manipulators: Cooperative and competitive cases. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 113–119. AAAI Press.

Garey, M., & Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

Gehrlein, W., & Lepelley, D. (2012). *Voting Paradoxes and Group Coherence: The Condorcet Efficiency of Voting Rules*. Springer.

Ghosh, S., Mundhe, M., Hernandez, K., & Sen, S. (1999). Voting for movies: The anatomy of recommender systems. In *Proceedings of the 3rd Annual Conference on Autonomous Agents*, pp. 434–435. ACM Press.

Hemaspaandra, E., & Hemaspaandra, L. (2007). Dichotomy for voting systems. *Journal of Computer and System Sciences*, *73*(1), 73–83.

Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2007). Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, *171*(5–6), 255–285.

Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2009). Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly*, *55*(4), 397–424.

Hemaspaandra, E., Hemaspaandra, L., & Schnoor, H. (2014). A control dichotomy for pure scoring rules. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pp. 712–720. AAAI Press.

Hemaspaandra, L., Lavaee, R., & Menton, C. (2013). Schulze and ranked-pairs voting are fixed-parameter tractable to bribe, manipulate, and control. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pp. 1345–1346.

Hemaspaandra, L., & Williams, R. (2012). An atypical survey of typical-case heuristic algorithms. *SIGACT News*, *43*(4), 71–89.

Kolliopoulos, S., & Young, N. (2005). Approximation algorithms for covering/packing integer programs. *Journal of Computer and System Sciences*, *71*(4), 495–505.

Lin, A. (2012). *Solving Hard Problems in Election Systems*. Ph.D. thesis, Rochester Institute of Technology, Rochester, NY.

Lu, T., & Boutilier, C. (2011). Budgeted social choice: From consensus to personalized decision making. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pp. 280–286.

McGarvey, D. (1953). A theorem on the construction of voting paradoxes. *Econometrica*, *21*(4), 608–610.

Meir, R., Procaccia, A., Rosenschein, J., & Zohar, A. (2008). The complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research*, *33*, 149–178.

Menton, C. (2013). Normalized range voting broadly resists control. *Theory of Computing Systems*, *53*(4), 507–531.

Menton, C., & Singh, P. (2013). Control complexity of Schulze voting. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 286–292.

Parkes, D., & Xia, L. (2012). A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 1429–1435.

Perek, T., Faliszewski, P., Pini, M., & Rossi, F. (2013). The complexity of losing voters. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pp. 407–414.

Rothe, J., & Schend, L. (2013). Challenges to complexity shields that are supposed to protect elections against manipulation and control: A survey. *Annals of Mathematics and Artificial Intelligence*, *68*(1–3), 161–193.

Russell, N. (2007). Complexity of control of Borda count elections. Master's thesis, Rochester Institute of Technology.

Waggoner, B., Xia, L., & Conitzer, V. (2012). Evaluating resistance to false-name manipulations in elections. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 1485–1491.

Wagman, L., & Conitzer, V. (2014). False-name-proof voting with costs over two alternatives. *International Journal of Game Theory*, *43*(3), 599–618.

Wojtas, K., & Faliszewski, P. (2012). Possible winners in noisy elections. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 1499–1505.

Xia, L. (2012). How many vote operations are needed to manipulate a voting system?. In *Proceedings (Workshop Notes) of the 4th International Workshop on Computational Social Choice*, pp. 443–454.

Xia, L., Lang, J., & Monnot, J. (2011). Possible winners when new alternatives join: New results coming up!. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 829–836. International Foundation for Autonomous Agents and Multiagent Systems.